Collaborative Cross-Layer Design in Wireless Sensor Networks

BY

MOHAMED SALEM HEFEIDA
B.Sc. (Arab Academy for Science and Technology, Alexandria, Egypt) 2004
M.Sc. (Arab Academy for Science and Technology, Alexandria, Egypt) 2006

THESIS

Submitted as partial fulfillment of the requirements
for the degree of Doctor of Philosophy in Electrical and Computer Engineering
in the Graduate College of the
University of Illinois at Chicago, 2013

Chicago, Illinois

To my parents, Ahlam and Salem, for being who they are.

ACKNOWLEDGMENTS

Throughout the course of this work, I had a unique opportunity to interact with some of the best scientists and researchers in the field, for which I am very grateful. First and foremost, I would like to express my thanks and appreciation to my advisor, Prof. Ashfaq Khokhar, for granting me enormous freedom in my research. Without his consistent support, patient guidance, and encouragement, this dissertation would not have seen the light. I would like to express my sincere gratitude to him for sharing his many years of academic and life experiences, which constitute invaluable lessons that will always accompany me wherever I go. Although many researchers know about Ashfaq's great achievements and contributions across many disciplines, very few get to experience and learn from his wonderful personality. A personality that exudes warmth among group members from very different backgrounds and working on very different projects.

I would also like to thank my thesis committee members, Prof. Rashid Ansari, Prof. Ajay Kshemkalyani, Prof. Milos Zefran, and Prof. Goce Trajcevski. Their encouraging comments, feedback, and stimulating discussions helped me improve and complete my Ph.D work. I would also like to take the opportunity to thank Prof. Aris Ouksel and Prof. Wenjing Rao for their useful discussions and suggestions.

I am indebted to Turkmen Canli, my friend and colleague. His support, feedback, and encouragement were invaluable. I enjoyed working with him very much, and learned that perfection is challenging but achievable. I would also like to thank Min Shen for his comments and interesting discussions.

I consider myself very lucky to be part of a research group with members from five continents. A great number of friends and colleagues have contributed to my life at UIC. I would like to mention, in no particular order, Heping Wang, Loay Abusalah, Xiabo Zhang, Hafiz Malik, Masroor Hussein, Muhammad Tahir, Xi Xu, Yasaman Keshtkarjahromi, Oliviu Ghica, Abinash Roy, Johnson Gad Elkarim, Mohamed Lodhi, and Hassan Janjua. I would in particular like to extend my deep thanks to my energetic friend and former lab mate, Fadi Almasalha (a.k.a Zaeem), for his company that ranged from research projects to fixing cars. Our common and diverse interests always kept us in touch outside the lab. To Fahad Saeed, for his amazing heart and beautiful mind. Our commonalities made us constantly doubt the fact that we came from two different continents. He is also the reason behind my constant cravings for Haleem and Nihari, two great dishes that he cooked very well. To Mohamed Ali, my dear friend and brother. Although he was mostly ”off campus”, we managed to get together often enough to discuss various topics from music and arts to religion and politics. He was also my first squash trainer, and he did it for free. To Umer Cheema, my very quiet and peaceful lab mate that has put up with a lot of philosophical discussions.

I would like to acknowledge the ECE administrative family for being very kind and helpful. I am especially thankful to Agustina Alvarado, Alicja Wroblewski, Beverly Miller, Erica Plys, Mihai Bora, Mona Hurt, and Nicholas Certa. I would also like to thank our wonderful lab staff for their help throughout the years. I am particularly thankful to Marta Salinas, George Ashman, and Harold Sosa.

It is always difficult for me to express my feelings, especially in writing. I have yet to find words that can express my thoughts and feelings towards my family. My role models and inspiration throughout my life and the reason behind any success, my beloved parents Ahlam and Salem. Just writing these lines bring tears to my eyes, you have been and will always be with me wherever I go. Your endless care for me and my siblings and tireless work have defined the meaning of life. My beautiful siblings, Khadiga, Asmaa, Fatma, ObaidAllah, and Yousef, your love and support are invaluable and I appreciate every moment we shared together. ObaidAllah and Yousef, you being next to mom and dad always eased off my sadness for my absence. My dear nieces and nephews, just watching you grow fills my heart with inexpressible joy, a joy that only a parent can experience. Lastly, but certainly not least, my wonderful wife Hawa. I deeply appreciate your patience and support along a path that had very few highs and many lows. I am grateful to your unparalleled care for me and our two beautiful children, Salem and Mariam, while I was busy with deadlines or trying to get some rest after a busy week. I deeply appreciate your endless efforts making up for my absence, and to your family's care and support.

MH

TABLE OF CONTENTS

TABLE OF CONTENTS (Continued)

LIST OF TABLES

LIST OF FIGURES

LIST OF FIGURES (Continued)

LIST OF FIGURES (Continued)

SUMMARY

Energy efficiency is a fundamental requirement in Wireless Sensor Networks (WSNs) due to its critical importance in a wide range of applications where recharging/replacing sensor batteries is unfeasible. In WSN applications such as field surveillance, environmental monitoring, disaster response, and traffic control, the cost of recharging batteries or replacing sensors may exceed the cost of deploying a new network. This led energy efficient operations to be the overarching goal of protocols employed at different layers of the network stack. In response, a plethora of research efforts have been pursued to achieve this goal, many of which utilize collaboration either between different layers of the network stack or between sensor nodes. In the literature, the former is known as Cross-Layer (CL) design and the latter is often referred to as Collaborative WSNs (CWSNs). Both CL design and CWSNs explore benefits of creating a new dimension of awareness by sharing information that is otherwise hidden and we refer to the former as intra-nodal collaboration and to the latter as inter-nodal collaboration. The benefits from such collaborations are mainly dependent on the context in which the new information is processed and utilized. Existing state of the art in this diverse area lack a unified design framework that allows seamless CL design for communication and processing operations in WSNs.

Towards this end, we propose several energy-efficient collaborative CL schemes with various contexts of interest. First, we develop CL-MAC, a Cross-Layer Medium Access Control protocol for synchronous WSNs. CL-MAC takes advantage of routing layer information and inter-nodal collaboration in order to efficiently handle multi-packet, multi-hop and multi-flow traffic patterns while adapting to a wide range of traffic loads. CL-MAC's scheduling is based on a unique structure of Flow Setup Packets (FSPs) that efficiently utilize routing information to transmit multiple data packets over multiple multi-hop flows. Unlike other MAC protocols, supporting construction of multi-hop flows, CL-MAC considers all pending packets in the routing layer buffer and all flow setup requests from neighbors, when setting up a flow. This allows CL-MAC to make more informed scheduling decisions, reflecting the current network status, and dynamically optimize its scheduling mechanism accordingly.

Second, we investigate supporting multi-hop and multi-packet routing in asynchronous WSNs in order to improve end-to-end latency and overall power consumption in such traffic patterns. We propose extending the knowledge of asynchronous MAC schemes to utilize a combination of routing information (intra-nodal) and duty-cycling information (inter-nodal) to temporarily synchronize nodes on the routing path between the transmitter and receiver. This idea can be applied to most asynchronous MAC schemes and was tested on RI-MAC (one of the recent energy efficient asynchronous MAC protocols).

Third, we study collaboration between MAC, routing and application layers in order to eliminate communication redundancy and improve load balancing across the entire network via forming data-dependent virtual clusters. We propose a CL Dynamic Virtual Clustering-based Data Gathering technique called DVC-DG. DVC-DG integrates overhearing at the MAC layer with data being processed/communicated at upper layers (i.e., routing and application layers) to realize implicit virtual clusters and eliminate data redundancy. Unlike most existing clustering-based data gathering solutions, which employ an explicit data-independent clustering algorithm to choose cluster heads and members, DVC-DG eliminates the need for special cluster head election mechanisms and distributes common centralized cluster-head responsibilities (e.g., data collection and aggregation) over all cluster members and eliminates data redundancy at its very source rather than at the cluster-head.

Finally, we propose a Cross-Layer Application-aware Paradigm (CLAP) that realizes and facilitates seamless collaboration across layers (intra-nodal) and between nodes (inter-nodal). CLAP allows any layer in the network stack to impact the behavior of other layers, according to the context of interest. The key is a new means of exchanging and processing CL information, called the Information-Layer (I-Layer). The I-Layer allows each layer of the network stack to publish its local information to be shared with other layers and subscribe other layers' shared information. Moreover, we augment CLAP into the SIDnet-SWANS simulator via a new API design which eliminates the need for hacking/bypassing conventional design hierarchies and simulator architectures. This greatly reduces the design and implementation complexities of CL protocols. Furthermore, to demonstrate CLAPs unique capabilities, we utilize it to develop a sample CL protocol, which constantly monitors the application's current demands (i.e., contexts of interest) and reconfigures underlying protocols accordingly.

CHAPTER 1

INTRODUCTION

Recent improvements in electronics and wireless communications have fueled the development of low power, low cost, highly integrated smart sensing devices. This promoted their usage in a broad range of application areas, such as military, law enforcement, industry, environment, and health. A Wireless Sensor Network (WSN) is composed of a number of tiny autonomous battery-powered smart sensor nodes deployed in a field. Each sensor node is capable of sensing its surrounding field, and processing and communicating sensory information. In addition to tiny size, low cost, and low power operation, the data processing capabilities of sensor nodes give WSNs distinctive cooperation characteristics. That is, nodes can perform computations locally, and then transmit processed data instead of raw data. Another main characteristic of many WSNs is that the position of sensor nodes need not be predetermined, which allows random deployment in harsh environments. On the other hand, this also requires WSN algorithms and protocols to exhibit self-organization.

The above unique characteristics attract a wide range of military and commercial applications, including battlefield sensing (e.g., mine detection, target tracking), environmental monitoring (e.g., habitat, disasters, traffic), and industrial sensing and control (e.g., hazardous situations, quality control). For example, in military applications, the self organizing and fault-tolerance (i.e., due to dense deployment) capabilities of WSNs makes them adequate for field surveillance and target tracking. While lack of infrastructure, low power decentralized operation, and scalability, are desirable characteristics that attract a wide variety of applications to utilizing WSNs, they also pose tremendous challenges on developing algorithms and protocols for such resource-constrained environments.

We summarize the main challenges, faced when designing and developing algorithms and protocols across a wide range of WSN applications, as follows:

- Resource constraints: considered the most critical in WSNs and include limited power supply, limited memory/computation capabilities, and limited wireless channel bandwidth. These constraints are mainly tied to the compact size and cost reduction requirements, which are main attributes that distinguish WSNs from traditional ad hoc networks.
- Adaptability/mobility: the environment, in which sensor nodes are deployed, may vary and cause variations in the sensed field, the wireless channel and may cause nodes to malfunction. This can cause significant fluctuations in traffic loads and patterns as the network topology and surrounding conditions change. These can a also result from node mobility.
- Load balancing: required in most applications in which all sensor nodes are identical, to ensure even load (i.e., communication and computation) distribution among all nodes. This results in even power consumption and thus helps prolong the overall network lifetime (i.e., the time elapsed before the battery of the first or last node in the network is depleted).
- Scalability: required in many WSN applications, in which sensor nodes are densely deployed and/or comprise a large scale network.

- Heterogeneity: many WSNs comprise of heterogeneous sensor nodes, and each node can incorporate multiple sensors. In fact, many commercially available motes integrate several sensors (e.g., temperature, pressure, humidity on a TELOSB mote platform (1)).
- Self-organization: needed when accurate positioning (during deployment or operation) of sensor nodes is not possible, therefore nodes are randomly placed. For example, in hostile environments, not only nodes are randomly deployed, they are also left unattended. This mandates node's self organizing capability in order to form a functioning network infrastructure, which usually does not pre-exist as in computer or telecommunication networks.
- Reliability/fault- tolerance: in shared communication media, channel interruptions may occur due to environmental interference and/or concurrent traffic flows in the network. This can result in transmission failures.

Each application requires a combination of the above features (challenges) and may add its own challenges, such as Quality of Service (QoS) requirements (2) (3). Not only different WSN applications will have different combinations of the above features, they may also differ in the priority of each. Moreover, a single application can go through phases of different priorities for each required characteristic. For example, in an industrial application, where the temperature of hazardous materials is being monitored, nodes sample the field and send the sensed value every ten seconds, as long as the sensed temperature is below a certain threshold. If the sensed value exceeds that threshold (indicating an emergency), nodes are required to sample the field and report their values every ten milliseconds. Under normal operating conditions (temperatures below threshold), this application may sacrifice data fidelity for extra energy savings, however, the opposite occurs when the threshold is exceeded. In bearing of the above challenges, without proper modifications, techniques adopted in the design and operation of traditional wireless networks (4) (5) are not suitable for resource-constraint networks. Therefore, special mechanisms need to be employed to tackle these challenges in the protocol designs, taking into account performance requirements and characteristics of the application. However, no single design is expected to optimally solve all the above challenges due to their complexity, possible conflicts, and dynamic application priorities. A good design should meet an application's major performance requirements. With the stringent energy constraints imposed by most WSN applications due to the infeasibility of battery recharging/exchanging, energy conservation is considered the utmost goal when designing protocols and algorithms for such resource-limited environments. This fueled the development of energy efficiency-oriented algorithms and protocols, many of which explore potential benefits from violating traditional design standards, which incorporate collaboration between different layers of the Open Systems Interconnection (OSI) model. Protocols that incorporate this type of collaboration are known as Cross-Layer (CL) protocols.

It is well known that communication operations consume orders of magnitude more energy compared to computation operations. For example, in a sensor node designed by Rockwell Inc., the ratio between the energy consumed in transmitting one bit to that consumed in processing a single CPU instruction is approximately 2000 (6). This steered the attention of many researchers to target communication operations. Reducing the cost of communication operations in WSNs can take many forms including:

1. avoiding main sources of energy waste (e.g., idle listening, overhearing, and collisions)
2. reducing communication overhead (e.g., control messages, and protocol overheads)
3. utilizing CL information (e.g., routing information, scheduling information, and application information)
4. in-network data processing (e.g., data aggregation, fusion, and compression)
5. in-network organization (e.g., network clustering, multi-hop, multi-packet, and multi-flow transmission scheduling)

As mentioned above, energy efficiency is considered the most critical challenge in many WSNs and can be tackled at different layers of the network stack (application layer $\rightarrow$ physical layer). However, since Medium Access Control (MAC) layer controls the most energy consuming component in the network (i.e., the radio), significant research studies have been devoted to design energy efficient MAC schemes (7) (8) (9). Many of these schemes rely on CL information to optimize their operation to meet specific application needs or adapt to varying operating conditions (e.g., variation in channel and network conditions). For example, in synchronous contention-based MAC schemes (see section 2.2), utilizing routing layer information to schedule packets over multiple hops in order to reduce end-to-end latency (10) (11) (12). In (13) and (14), the MAC information is utilized at the routing layer to achieve minimum end-to-end latency routes. Although these studies achieve improvements in terms of energy-efficiency, end-to-end latency, and throughput, none of them responds to the overall global status of the node/network.

1.1 Contributions

In light of the above challenges and potential gains, this dissertation focuses on improving energy-efficiency in a wide range of applications and network scenarios, taking into consideration other major challenges such as, end-to-end latency, throughput, network/node heterogeneity, load balancing, data redundancy, scalability, and design complexity. At its core, each proposed solution utilizes CL interaction (a.k.a. intra-nodal collaboration) and inter-nodal collaboration. We develop various algorithms and protocols that significantly reduce energy consumption while improving classical performance metrics such as end-to-end latency, network lifetime, and throughput. We also propose a design paradigm to facilitate collaboration across layers (intra-nodal) and between nodes (inter-nodal). The main contributions of this thesis can be summarized as follows:

1. Developing CL-MAC, a Cross-Layer MAC protocol for synchronous heterogeneous WSNs. CL-MAC utilize routing layer information and inter-nodal collaboration in order to efficiently handle multi-packet, multi-hop and multi-flow traffic patterns while adapting to a wide range of traffic loads. Significantly different from other MAC protocols, supporting construction of multi-hop flows, CL-MAC considers all pending packets in the routing layer buffer and all flow setup requests from neighbors, when setting up a flow. This allows it to make more informed scheduling decisions, reflecting the current network status, and dynamically optimize its scheduling mechanism accordingly (15).

2. Developing a CL technique to support multi-hop and multi-packet routing in asynchronous WSNs, in order to improve end-to-end latency and energy efficiency in such scenarios. The proposed solution temporarily synchronizes nodes on the routing path between transmitter/receiver pairs. It is generic and can be applied to many asynchronous MAC scheme (16).

3. Developing a CL data gathering method that exploits collaboration between MAC, routing and application layers in order to eliminate communication redundancy and improve load balancing across the entire network via forming data-dependent virtual clusters. Unlike many existing clustering-based solutions, our method eliminates the need for special cluster-head elections and distributes centralized cluster-head responsibilities among all cluster members. It also eliminates data redundancy at its very source rather than at the cluster-head (17).

4. Developing a CL design paradigm that realizes and facilitates collaboration across layers and between nodes. The proposed paradigm allows any layer in the network stack to impact the behavior of other layers, according to the current context of interest. The key is a new means of exchanging and processing CL information. We augment the proposed paradigm into a network simulator via a new API design which eliminates the need for hacking/bypassing conventional design hierarchies and simulator architectures (18) (19).

5. Demonstrating the capabilities of the proposed CL design paradigm by utilizing it to develop a CL application-controlled protocol, which constantly monitors the application's current demands and reconfigures underlying protocols accordingly (19).

1.2 Thesis Organization

Chapter 2 starts with a brief introduction to latency, throughput and energy efficiency problems in the context of multi-hop traffic, particularly in heterogeneous WSNs. It briefly discusses previous cross-layer MAC protocols with a focus on those utilizing routing information to setup multi-hop flows in synchronous duty-cycled WSNs. Then it discusses the details of a CL solution (CL-MAC), aiming at facilitating the transmission of multiple multi-hop packets over multiple flows. We also validate CL-MAC's performance with comparison to state of the art. Chapter 3 extends the idea of supporting multi-hop traffic to asynchronous duty-cycled WSNs. We develop a CL design approach for asynchronous MAC schemes to support transmission of multiple-packets over multiple-hops and hence improve latency and throughput of asynchronous WSNs. The proposed approach utilizes routing information at the MAC level and communicates duty-cycling information to temporarily synchronize next hop nodes. We study the cost of sharing information within a node (between layers) as well as between nodes, which is reflected in extra memory requirements and communication overhead, respectively. Chapter 4 presents a CL data-dependent Dynamic Virtual Clustering-based Data Gathering (DVC-DG) technique that incorporates an Overhearing-dependent contention-based MAC (OD-MAC) scheme. The proposed technique eliminates the need for special cluster head election mechanisms and ensures load balancing among all nodes. We analytically demonstrate the energy efficiency of the proposed approach, and results show a reduction in the number of communication operations by a factor of up to N (number of nodes in a neighborhood), compared to existing state of the art. In Chapter 5, we propose a Cross-Layer Application-aware Paradigm (CLAP), to facilitate

CL protocol design in WSNs and overcome implementation complexities encountered in most CL approaches. This introduces a new level of application layer awareness and control over underlying protocols, which is a key distinction between CLAP and other CL approaches. A sample application scenario is also presented to show the effectiveness of the proposed paradigm. Chapter 6 concludes the thesis and points out future directions in the field.

CHAPTER 2

CL-MAC: A CROSS LAYER MAC PROTOCOL FOR HETEROGENEOUS SENSOR NETWORKS

In this chapter, we give a brief introduction to the latency, throughput and energy efficiency problems in the context of multi-hop traffic, particularly in heterogeneous WSNs. We briefly discuss previous related work in synchronous duty-cycled MAC protocols with a focus on those utilizing routing information to setup multi-hop flows. We present CL-MAC's design details, including flow setup and data transmission. We also evaluate CL-MAC's performance via extensive NS-2 simulations, comparing its performance to state of the art in its category of MAC protocols.

2.1 Introduction

Recent advances in Very Large Scale Integration (VLSI) and Micro-Electro-Mechanical Systems (MEMS) have fueled the development of low power, low cost, highly integrated sensing devices. This enabled the design and development of a wide variety of WSN applications such as target tracking, surveillance, environmental monitoring, and medical systems (7) (8) (20) (21). Many of these applications are increasingly employing heterogeneous sensor nodes. In fact, many commercially available motes integrate several sensors (e.g., temperature, pressure, humidity on a TELOSB mote platform (1)).

In heterogeneous WSNs, where a single node integrates several sensors (or different types of nodes networked together), each of which monitors a different phenomenon, variations in traffic loads and patterns are not tied to a single field's temporal and/or spatial correlation (22) (23). This can often result in conflicting scheduling demands in response to simultaneous different variations in independent fields. That is, two uncorrelated sensed values are required to be reported by the same node (or two neighboring nodes), at the same time (24). This requires MAC schemes to handle multiple flows of different data rates (i.e., from different sensors on the same node) while sustaining their main task of minimizing energy consumption, latency and maximizing throughput.

Duty cycling was introduced in (25), to reduce energy consumption due to idle listening, and was adopted in most of the following MAC protocols designed for WSNs (7) (8) (26) (27) (28). In duty cycled protocols, nodes go to sleep periodically to save energy. In such protocols, the sleep period cannot be utilized even if nodes have packets to send. This is clearly a trade-off between energy consumption and end-to-end latency which is directly proportional to the distance between source and destination.

To address the end-to-end latency problem of duty cycled MAC protocols in multi-hop packet transmissions, CL routing supported MAC schemes that set up multi-hop flows using routing layer information, such as (10) (11), have been proposed. In order to reduce the end-to-end delay and utilize the sleep period more effectively, these protocols set up multiple hop flows in a single cycle. Even though end-to-end latency was reduced in (10) (11) and similar studies (12), they suffer from:

1. Poor adaptation to heavy traffic loads,
2. Inefficient handling of multiple packet transmission within a single flow (even if nodes have additional packets to send),
3. Lack of support for multiple flow construction from a single node (creates bottle necks at nodes lying at the intersection of different flows),
4. Limited utilization of available routing information.

The main reason for the above shortcomings is the *single packet based* flow setup. More specifically, the flow setup is designed with a focus on single packet transmission over a single flow. These limitations become more critical if the underlying application employs heterogeneous sensors, as discussed above.

To address the above limitations, we propose a Cross-Layer MAC protocol (CL-MAC) that adapts to greatly varying traffic loads and is capable of simultaneously handling multiple multi-packet flows. Most importantly, CL-MAC can schedule multiple packets over *multiple multi-hop flows* in a single cycle. This is achieved by considering *all* pending packets in the routing layer and all pending flow setup request(s), when scheduling channel allocation. As a result, MAC layer can detect traffic load variations, and reserve time for transmission of packets accordingly.

In CL-MAC, nodes setup communication via uniquely structured flow setup packets (FSPs) capable of addressing multiple destinations. The length of an *FSP* at any given node depends on the number of flows it is part of, the node's position in the flow(s), the number of packets it has to send, the number of receivers it sends to, and the remaining time in the data period. All this information is available via the routing layer. This allows CL-MAC to have a better assessment of the current traffic load/pattern status and hence make a more informed decision when setting up a flow.

In addition, CL-MACs control overhead is more *efficient* compared to other synchronous contention-based protocols with simpler scheduling schemes, such as S-MAC (25). This is due to the ability of an *FSP* to address multiple destinations while simultaneously representing multiple packets. This greatly reduces control overhead compared to other CL single packet/flow oriented protocols (10) (11).

To the best of our knowledge, no other protocol has the capability of adapting to greatly varying traffic loads, network topologies and supporting scheduling of multiple packets over multiple multi-hop flows, as explained above. Our contributions can be summarized as follows:

1. We develop a novel CL duty cycled MAC protocol, CL-MAC, for heterogeneous and homogeneous WSNs.
2. CL-MAC incorporates a unique efficient flow setup scheme capable of scheduling multiple multi-hop flows, each flow consisting of multiple packets, all in a single cycle.
3. CL-MACs new flow setup capabilities originate from considering all buffered data packets in addition to routing layer information, when setting up flows.
4. CL-MAC minimizes control overhead by having Flow setup packets *(FSPs)* address multiple destinations.
5. CL-MAC dynamically adapts to a wide spectrum of traffic patterns and a very large range of non-uniform traffic loads while significantly improving delivery ratio, end-to-end latency, without compromising energy efficiency.

2.2 Related Work

Energy-efficient MAC protocols for sensor networks can be generally divided, based on the channel access technique, into three main categories: contention-based, contention-free, and Hybrid. Contention-based MAC protocols can be further divided into synchronous and asynchronous. In synchronous contention-based protocols (like CL-MAC), nodes coordinate their wakeup and sleep periods and communicate while awake (e.g., (25) (26) (28) (29). While this category of protocols requires periodic synchronization, it is much looser compared to contention-free MAC protocols, which require much tighter synchronization in addition to slot allocations (30) (31). On the other hand, in asynchronous contention-based protocols, there is no coordinated sleep/wakeup process. Instead, the sender communicates with the receiver by sending an alert message containing the receiver's ID (preamble) before it sends the actual data packet (32) (33) (34) (35). Although simple to implement, asynchronous schemes can not provide guarantees on the worst-case delay. Finally, hybrid MAC solutions combine features from both contention-based and contention-free MAC protocols (36) (37) (38). Despite combining desirable features from the other two categories, hybrid protocols also suffer from combined complexities (i.e., periodic synchronization and slot allocations). More details about MAC protocol classifications, characteristics, and limitations can be found in (7) (8) (21).

Cross-layer MAC protocols can belong to any of the above categories. However, those utilizing routing information to improve end-to-end latency in multi-hop duty cycled networks can greatly benefit from some type of synchronization/scheduling. This is present in contention-free schemes, synchronous contention-based schemes, and hybrid schemes (8). Although contention-free MAC schemes (e.g., TDMA) provide collision-free channel access and can schedule node wakeups, they suffer from slot allocation and management requirements in addition to tight synchronization demands. These requirements are also present in hybrid protocols, but with looser synchronization constraints. Synchronous contention-based MAC protocols also require loose synchronization, however, no slot assignment/management is required. This makes them more scalable and simpler to implement, which is desirable in many WSN applications (7) (8).

Several CL protocols, from different categories, studied the utilization of routing layer information to address the end-to-end latency problem of duty cycled MAC protocols in WSNs (10) (11) (12) (28) (31) (36) (37) (39) (40) (41) (42) . In this chapter, we focus on synchronous contention-based MAC solutions (10) (11) (12) (39) (40) (41). Although these studies propose viable solutions for the problem, none of them utilizes routing information as CL-MAC does nor matches its scheduling capabilities highlighted in section 2.1.

Routing enhanced MAC (RMAC) (10) and Look-Ahead Scheduling MAC (LAS-MAC) (41) are among the first studies that proposed multi-hop flow set-up for WSNs, assuming a single packet per flow. RMAC sets up multi-hop flows using Pioneer frames (PIONs) that replace the common Request To Send (RTS) and Clear To Send (CTS) packets, such that to the upstream node a PION means CTS and to the downstream node it means RTS. A PION packet includes addresses of the final destination node, PION sender (source), next hop (destination), previous hop (the node that sent the PION packet to the current sender), and the order of the PION sender in the flow. LAS-MAC differs from RMAC in how the last node in a flow responds to flow setup requests, how nodes acknowledge data packets, and how collisions are handled.

In both RMAC (10) and LAS-MAC (41), if a source node or intermediate nodes have packets to send to the final destination of the flow, additional flow setups are required for each data packet. The limitation of single packet per flow policy is the extra control packets and packet delays. Pipelined-RMAC (PRMAC) (12) extends RMAC (10) to allow transmission of multiple packets in a constructed flow. Flow setup is based on (10), however, during the flow setup, nodes negotiate with their previous and next hop neighbors in the flow to decide how many packets they can receive and send. To facilitate this negotiation, two fields are added to the PION packet, packetsToSend and packetsToRecv. Adaptive Multi-hop MAC (AM-MAC) (43) improves RMAC's performance by reducing energy wasted in idle listening via adjusting the duration of the listening period according to the traffic load. In addition, to reduce control overhead, AM-MAC employs a new control packet that replaces the RTS/CTS pair during a multi-hop transmission and DATA packets can serve as acknowledgments.

Demand Wakeup MAC (DW-MAC) (11) schedules data packet transmissions using scheduling frames (SCHs) in lieu of SMAC's RTS/CTS packets and RMAC's PION packets. For unicast packets, an SCH includes sender, receiver, and final destination addresses, and transmission duration. For broadcasts, an SCH includes the sender's address and sequence number. DW-MAC is different from RMAC in several aspects. First, DW-MAC allows multiple nodes (that are neighbors) to send data packets within one cycle. Second, DW-MAC supports broadcast. Third, an SCH packet does not include timing information like RMAC's PION packet, i.e., PION packet includes the order of the node. Finally, data transmission time is embedded in the SCH position in the data period. Adaptive Scheduling MAC (AS-MAC) (39) is based on DW-MAC (11), however, the duration of data periods are variable, and depend on the traffic load. The variable duration data period allows longer multi-hop flow setup when needed. Moreover, when the traffic load is light, nodes can go to sleep without waiting for the whole data period, hence saving energy.

To the best of our knowledge, our preliminary work "BulkMAC" was the first CL protocol to support multiple flow setup during a single cycle (40). This feature is particularly useful at nodes falling in the intersection of multiple flows and has shown great reduction in collisions and delay (40). BulkMAC also extended the upper limit on the maximum number of packets that can be sent in a single cycle via a smart scheduling mechanism that accesses the routing buffer and differentiates between packets based on their destinations.

CL-MAC extends BulkMAC's scheduling capabilities by scheduling multiple packets over multiple *multi-hop* flows in a single cycle. That is, multi-flow setup is not restricted to a single hop as in BulkMAC (40). CL-MAC also extends BulkMAC's single flow multi-hop scheduling capability via its new flow setup mechanism. The flow setup mechanism in CL-MAC is based on pending packets in the routing layer, and *pending flow setup requests.* This allows nodes to make more informed decisions when setting up a flow, compared to other CL protocols which set up their flow based on a single packet and/or ignore other nodes' flow setup information (10) (11) (39). As a result, CL-MAC can adapt to variations in traffic and reserve segments accordingly, in a single cycle. CL-MAC also incorporates an early acknowledgment scheme which allows a node to accommodate more flow setup requests in a single cycle.

2.3 CL-MAC Design

In this section we illustrate the design details of CL-MAC. We start with an overview of the proposed protocol, followed by a detailed discussion of control packets, Network Allocation Vector (NAV) reservation policy, flow setup scheme, early acknowledgment procedure and conclude with data transmission methodology.

2.3.1 Overview

In CL-MAC, nodes setup communication via flow setup packets (FSPs). An *FSP* serves as a request to send (RTS) packet to the destination node and a clear to send (CTS) packet to the source node, as in (11) (10). However, due to its unique structure, only *FSPs* have the ability to address several destinations i.e., one *FSP* can operate as an RTS for up to K different destinations.

The length of an *FSP* at any given node depends on:

1. the number of flows.
2. its position in the flow.
3. the number of packets it has to send.
4. the number of receivers it sends to.
5. the remaining time in data period.

The timing of an *FSP* packet determines the reserved segment's position in the sleep period as in (11). Figure 1 shows how an *FSP's* timing determines the reserved time in the sleep period. The first segment in FSP_i is used for the first destination, the second segment is reserved for

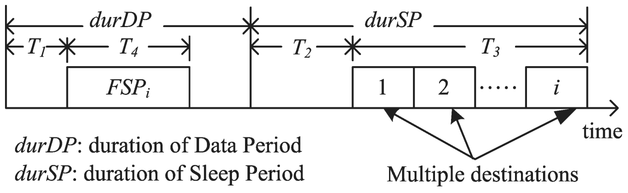

durDP
durSP
T1
T4
T2
T3
FSPi
1
2
i
time
durDP: duration of Data Period
durSP: duration of Sleep Period
Multiple destinations

Figure 1. FSP Timing Diagram: FSP_i reserves i contiguous segments in the sleep period

the second destination, and so on. Equation 2.1 gives the start time of the segment reserved for sending data packets to the i^{th} destination in FSP_i.

$$T_i = T_2 + (i-1) \times R \times |FSP_1| \tag{2.1}$$

and

$$R = \frac{T_2}{T_1} = \frac{T_3}{T_4} = \frac{durSP}{durDP} \tag{2.2}$$

The multi-hop, multi-destination, and multi-packet scheduling ability of CL-MAC can be illustrated by the following example. In the network setup shown in Figure 2(a), suppose that node A has two packets for node C and two packets for node E. Furthermore, node A sends packets to node C and node E through node B and node D, respectively. Nodes X and Y are potential transmitters and node Z is a potential receiver.

If node A wins the contention, it sends an FSP_2 packet with the first destination address set to node B, and the second destination address set to node D. The first destination, node

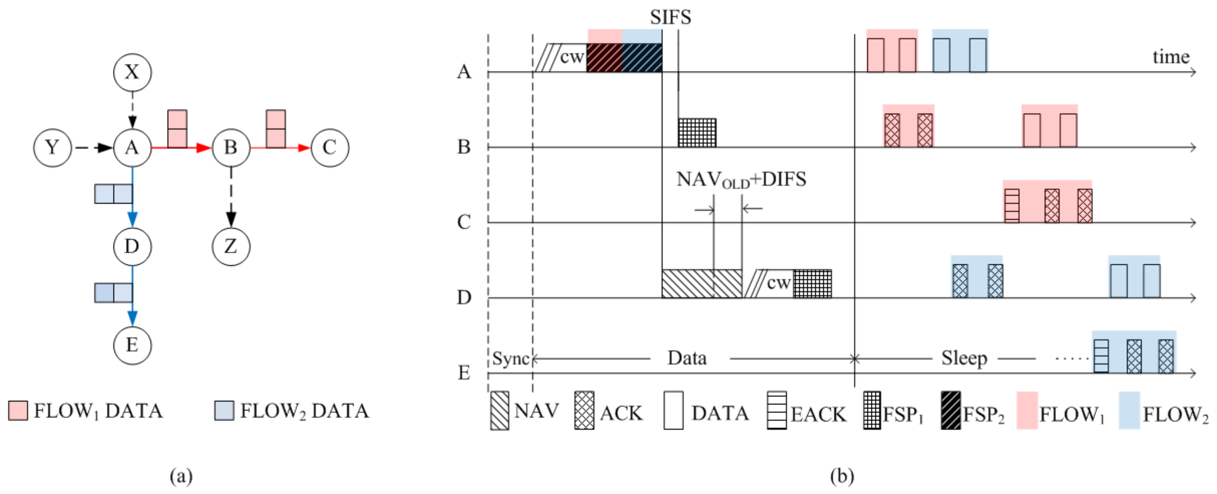


Figure 2. CL-MAC Flow Setup: (a) Sample Network Setup, and (b) Scheduling Details

B, sends an FSP_1 to node C after SIFS seconds. Node D, reserves the following $|FSP_1| + |SIFS| + |DIFS|$ seconds in its NAV, and starts contending at the end of this period. As shown in Figure 2(b), if node D acquires the channel, it sends an FSP_1 to node E. If node B had pending packets to node Z, after receiving node A*'s* FSP_1, it could have sent an FSP_2 to nodes Z and C to setup a multi-destination flow. Note that the final destination nodes, nodes C and E, do not send any acknowledgment packets during the data period. CL-MAC uses an early acknowledgment mechanism to allow longer multi-hop flow setups and more scheduling of packet transmissions during the data period which will be explained in detail in section 2.3.6.

Figure 3 illustrates the diverse packet scheduling and flow setup capabilities of CL-MAC. Assume that a node can send λ packets in a single reserved segment in the sleep period. At one

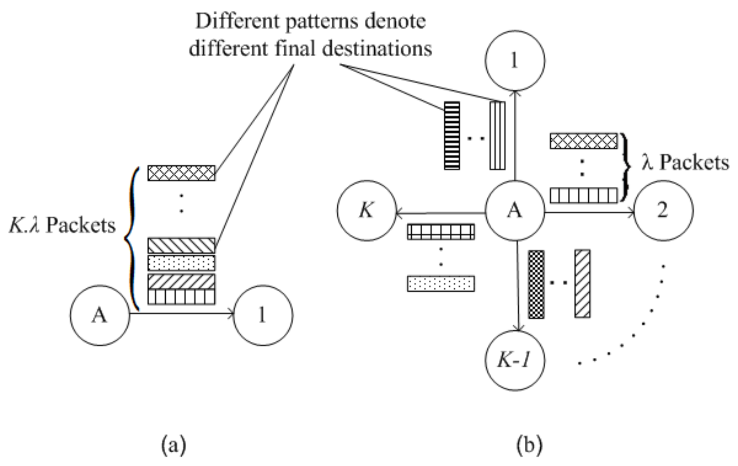


Figure 3. Representing CL-MAC's diverse flow setup capabilities: (a) $K \times \lambda$ packets to next hop, and (b) λ packets to K one hop away destinations

extreme, a node can send $K \times \lambda$ packets to its next hop in one cycle as shown in Figure 3(a). At the other extreme, within a single cycle, a node can schedule one packet to up to $K.\lambda$ of its neighbors (Note that this is not a broadcast since each packet is different). Figure 3(b) illustrates an intermediate scenario between both extremes, where a node A has λ packets specific to k of its neighbors, it can schedule $one-to-K$ specialized packet transmissions, and it can send up to $K \times \lambda$ packets in total, all within a single cycle. Note that if the destinations are N hops away, then each can only receive up to λ/N packets. These capabilities result from a set of key design aspects discussed in the remainder of this section.

2.3.2 Flow Setup Packets

CL-MAC uses flow setup packets *(FSPs)* to schedule data packet transmissions. There are K different *FSP* types supported in CL-MAC, the main difference between each type is in the reservation capability, i.e., the number of unit segments they can reserve in the sleep period.

An *FSP* of type i is denoted by FSP_i, reserves i unit segments in the sleep period. The duration of FSP_i is i times the duration of an FSP_1. The reserved time segment's start and end times depend on the duration and relative sending time of the FSP_i. In other words, the time interval in which the FSP_i is sent in the data period is mapped to a time segment in the sleep period. Mapping is one-to-one and the mapping function uses the ratio of duration of sleep period to duration of data period, R, given by equation Equation 2.1.

Suppose node A sends an FSP_i t seconds after the data period starts, then the reserved time segment for the j^{th} destination in FSP_i , denoted by Π_A^j, is calculated as follows:

$$\Pi_A^j = [T_{SLEEP} + R(t + (j-1)|FSP_1|), T_{SLEEP} + Rt + Rj|FSP_1|] \tag{2.3}$$

In a single reserved time segment, the maximum number of packets a node can send is:

$$\lambda = \lfloor \frac{R \times |FSP_1|}{u} \rfloor \tag{2.4}$$

An FSP_i, where $1 \leq i \leq K$, is an RTS for up to i unique destinations, and to the previous node (upstream node), it is a CTS message. An FSP_i includes i final destination - next hop address pairs, source node address, and the address of the previous node. Note that, if node

TABLE I

NOTATIONS USED IN DESCRIBING CL-MAC FUNCTIONALITY

Symbol / Abbreviation	Description
R	Duration of sleep period over duration of data period.
u	Duration of time required to complete a single data packet transmission from one node to another.
T_R	Duration of reserved segment.
T_{SLEEP}	Start time of sleep period.
FSP	Flow setup packet.
T_{SLEEP}	Start time of sleep period.
μ_S	Receive start timeout.
μ_R	Receive start timeout.
Π_A^j	Time segment reserved by node A to the j^{th} node in FSP in sleep period.
λ	Maximum number of packets that can be transmitted in the minimum reserved time segment.
$TBSL$	To Be Sent List.
DSL	Dont Send List.
FSP	Flow Setup Packet.
$EACK$	Early Acknowledgment
$DIFS$	Distributed Inter-frame Space.
$SIFS$	Short Inter-frame Space.
NAV	Network Allocation Vector.

A has more packets to deliver to its prospective downstream node B than it can send within a unit segment, it puts node Bs address multiple times adjacent to each other in the next hop field in order to reserve more segments for node B in the sleep period.

2.3.3 Network Allocation Vector

In CL-MAC, nodes access to the wireless medium during the data period is contention-based as in (25). However, Network Allocation Vector (NAV) reservation times favor the construction of *longer multi-hop flows* by making nodes later in the destination field wait longer. This is based on the fact that packets of shorter paths suffer less end to end delay than those having longer paths to travel.

Nodes set their NAV according to (a) their relative order in the destination field in the FSP packet, provided that the node is one of the intended destinations, and (b) the type of FSP packet received, if the node is not one of the intended destinations.

NAV reservation details in CL-MAC can be further illustrated using the following example. When a node A receives an FSP_y, and it is not one of the y destinations, it will prevent itself from accessing the channel for the next RT seconds, as calculated in equation Equation 2.5 by updating its NAV.

$$RT = y(|FSP_1| + SIFS) + DIFS \tag{2.5}$$

On the other hand, if node A is the i^{th} destination, $i > 1$, then it will reserve the next RT_i seconds in its NAV as calculated in equation Equation 2.6. Table I includes a list of notations used in illustrating CL-MAC design and operation.

$$RT_i = (i-1)(|FSP_1| + SIFS) + DIFS \tag{2.6}$$

2.3.4 Flow Setup

In this section, we describe in detail how a node compiles a flow setup request to more than one destination, how it decides the list of neighbors it is going to send an FSP to, how many segments it reserves per destination, and how flow requests are acknowledged.

Similar to other scheduled based MAC protocol, the time is divided into frames or cycles. Within each frame, CL-MAC allows construction of multiple flows from a single node and supports transmission of multiple packets over multiple hops. During the setup phase, any flow can branch out if either the first node or intermediate nodes have pending packets to different destinations than the final destination(s) in an *FSP*. This gives CL-MAC its unique adaptability to different traffic patterns. In other words, the first node of the flow can start a flow having up to K different final destinations which may be multiple hops away, or intermediate nodes of the flow may add additional final destinations if they have pending packets for them. Moreover, in each reserved segment, a node can send multiple packets.

An important aspect of CL-MAC's flow setup is how a node decides the *FSP* type, i.e., how many segments it is going to reserve, and how it decides to which neighbors it should send a flow set up request. When a node receives a flow set up request, it will not confirm the request during the data period if (a) it is the final destination, or (b) there is no time left in the current frame for sending an *FSP* packet.

A CL-MAC node determines the type of *FSP* and the destination nodes as follows:

1. It computes the largest *FSP* (FSP_m, where $1 \leq m \leq K$) it can send during the remaining time of the data period.
2. It will reserve $\hat{m}$ segments, for the pending requests, where $\hat{m}$ is the number of segments requested by the pending requests ($\hat{m} \leq m$). Note that each pending request may require multiple segments.

3. Remaining $m - \hat{m}$ segments, provided that $m - \hat{m} > 0$, are used for pending packets in its routing layer.

4. *FSP* type is equal to the total number of segments used for pending requests and pending packets in the routing layer. If two segments are used, then *FSP* type will be FSP_2.

To prevent multiple requests via separate *FSP* packets from one node to another in the same data period, nodes keep a *dont send list* (DSL) of their neighbors that the node does not wish to send. This list is empty at the beginning of each data period (i.e., the list is cleared at the beginning of each data period), and as nodes send *FSPs*, they update their DSL by putting the address of every neighbor they have sent an *FSP*. As a result, a node makes at most one request to each one of its neighbors in any given cycle.

The reservation of segments for the pending packets in routing layer can be explained as follows. After reserving $\hat{m}$ segments to pending requests, the number of remaining segments is $m - \hat{m}$. First, MAC layer gets a *list* of final destinations of *pending packets* (PPL) from the routing layer. Second, it removes the entries in the PPL that need to be routed through any node in the DSL, this filtered list is referred to as the *to be sent list* (TBSL). Third, it starts assigning the remaining $m - \hat{m}$ segments to the final destinations in the TBSL until it becomes empty or there are no remaining segments in the data period. The first step of the assignment process can be explained as follows. MAC layer assigns the $(\hat{m} + 1)^{th}$ segment to the first final destination in the TBSL (TBSL[1]), and removes the next $\lambda - 1$ entries in the TBSL that are equal to TBSL[1], these $\lambda - 1$ packets will be sent in the $(\hat{m} + 1)^{th}$ segment. Moreover, the

MAC layer will remove the final destination addresses that are not equal to TBSL[1], but have to be routed through the same next hop node of TBSL[1].

2.3.5 Deferred Confirmation

The data period is used for scheduling packet flows, and it is important to reduce control traffic during this period as much as possible to increase the possible number of flow setups. CL-MAC introduces a delayed confirmation technique in which receiver nodes send an early acknowledgment *(EACK)* packet to their prospective sender nodes at the beginning of the segment reserved for them by their prospective senders.

The deferred confirmation technique is used in two different scenarios to reduce control packet traffic. First, each segment in the *FSP* has a destination and final destination fields associated with it. On reception of an *FSP*, the final destination node does not confirm the request in the data period. Instead, it will send an early acknowledgment in the sleep period. This policy enhances the sink node's capability of accepting more flow set up requests in one cycle. Second, a node may not confirm all of the flow set up request(s) it has received during the data period. In this case it will send an *EACK* packet to its requesters. Note that if a flow setup requester reserves more than one segment, the receiver node will only send an EACK at the beginning of the first segment. On the sender's side, if a node does not receive confirmation from the nodes it has requested flow setups from, it will assume that these nodes will send *EACKs* during the sleep period.

2.3.6 Data Transmission details

At the end of the data period, every node knows from which nodes it is going to receive data packets, the length of the reserved segment, and whether it should send an EACK to sender nodes or not. Also, it knows which nodes it's going to send data packets to, the number of segments it's going to use, and whether it should expect EACKs from receiver nodes.

Four possible data transmission scenarios are illustrated in Figure 4. In the first scenario, as shown in Figure 4(a), the sender node has received confirmation from the receiver, hence, it starts sending at the beginning of the reserved segment. Note that, the receiver does not know the number of packets it will receive, hence, if it does not start receiving data packets within μ_R seconds after its last acknowledgment, it goes to sleep. The sender node simply goes to sleep after receiving an acknowledgment if it does not have more packets to send.

In the second scenario, as illustrated in Figure 4(b), neither the sender receives a confirmation to its request, nor does the receiver send a confirmation to the sender node. At the beginning of the reserved segment, the receiver node sends an *EACK*, and the sender node starts sending data packet(s).

In the third scenario as shown in Figure 4(c), the receiver sends a confirmation to the sender but the sender does not receive it (possibly due to packet collision). In this case, if a receiver does not start receiving data packets within μ_R seconds from the beginning of the reserved segment, it will infer that its confirmation is not received by the sender, and sends an *EACK*.

Finally, in the fourth scenario, as illustrated in Figure 4(d), the sender has sent a request to the receiver node, but the receiver has not received it. In this case, the receiver does not

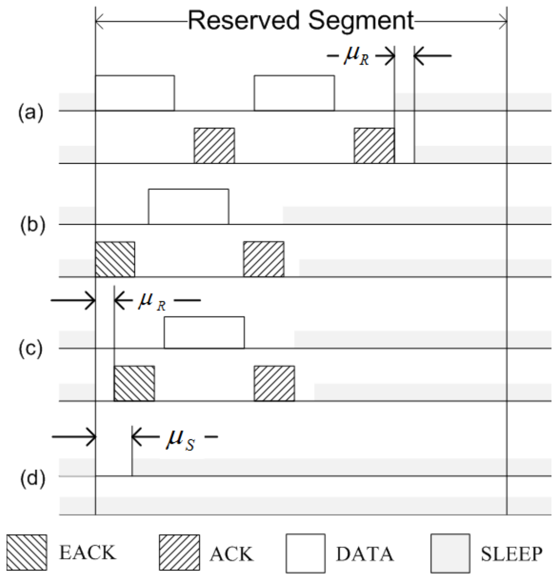


Figure 4. Data transmission scenarios: (a) sender received confirmation from receiver, (b) neither sender receives a confirmation nor receiver sends one, (c) receiver sends confirmation but sender does not receive it, and (d) receiver does not receive sender's request

wake up, and the sender waits for an $EACK$ from the receiver. If it does not receive an $EACK$ within μ_S seconds, it will assume that its request is lost, and goes to sleep.

2.4 Performance Evaluation

In this section we evaluate CL-MAC's performance using the NS-2 simulator (44). Each node in our simulations has a single omni-directional antenna and follows NS-2's commonly used combined free space and two-ray-ground reflection propagation model for wireless sensor networks. We also assume that nodes follow a single sleep/wakeup schedule and routing information is available to them. Key network simulation parameters are summarized in Table II. The transmission range and carrier sensing range are modeling a 914MHz Lucent WaveLAN DSSS radio interface which was used in several previous studies including (10) (11). All pro-

TABLE II

SIMULATION PARAMETERS (CL-MAC)

Parameter	Value	Parameter	Value
Bandwidth	20kbps	*Comm. Range*	250m
RxPower	0.5W	*Interference Range*	550m
TxPower	0.5W	*DIFS*	10ms
IdlePower	0.45W	*SIFS*	5ms
SleepPower	0.05W	*Contention Window*	64ms
$PION/SCH$	14B	*Cycle (RMAC)*	4725ms
FSP_1	12B	*Cycle (DW-MAC)*	4745ms
DataPckt	100B	*Cycle (BulkMAC)*	4544ms
$EACK$	10B	*Cycle (CL-MAC)*	4324ms
$SYNC$	55.2ms	*Data(RMAC)*	168ms
$Data(Bulk/DW)$	181ms	*Data (CL-MAC)*	161ms

tocols follow a 5% duty cycle and assume all nodes are already synchronized; therefore no synchronization traffic is included (applied for all simulated protocols for fairness). In our simulations, data packets are 100 bytes and various traffic loads are generated using different constant bit rates. We compare CL-MAC's performance against RMAC (10), DW-MAC (11) and BulkMAC (40), which were also implemented in NS-2, under various traffic loads and patterns, and different network topologies.

MAC protocols designed for homogeneous WSNs are generally tuned for low traffic rates, and their performance under high non-uniform traffic loads are often not explored. However, heterogeneous WSNs can experience a wide range of such traffic loads (7) (8) during a single frame (i.e., one sensor simultaneously generating data at ten times the rate of another sensor).

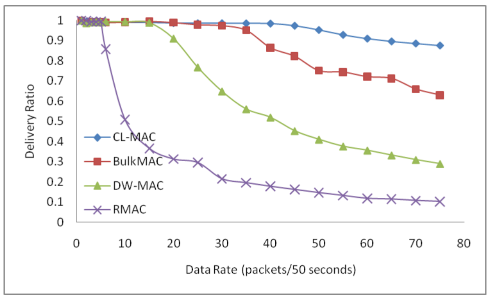
Therefore, we study the performance of MAC protocols over a wide range of traffic loads to show their adaptability to such circumstances.

The main performance metrics we're interested in are delivery ratio, end-to-end delay, energy consumption, and energy consumed/packet delivered. *Packet delivery ratio* is the ratio of the number of packets delivered to their destinations to the total number of packets generated. *End-to-end delay* is the time a packet takes from its source to its final destination. *Energy consumption* is the energy consumed by each node during the simulations. Finally, *energy consumed/packet delivered* is the energy consumption divided by delivery ratio. All the above metrics are averaged over the simulation duration as illustrated below for each scenario.

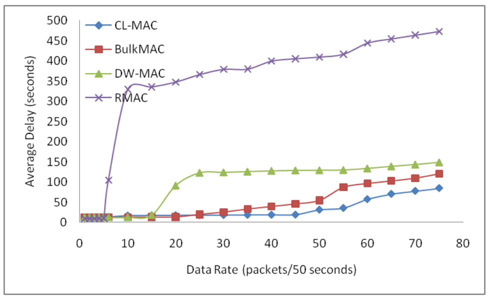
We compare CL-MAC's performance to (10) (11) (40) on a 15-node chain network, a 25-node 12-hop cross chain network, and a 100-node randomly connected network. On both chain and cross-chain networks, the distance between two adjacent nodes is 200 meters and data rate is increased from 1 packet/50 seconds to 75 packets/50 seconds. The traffic flow is generated by the node at one end of the network send packets to the node at the other end. However, on the cross chain network, each source generates traffic at a randomly chosen rate with the average of both rates reported in the results. On the random network, the percentage of active source nodes is increased from 5% to 45%, every source node generates data packets at a randomly chosen rate varying from one packet/10 seconds to 1.5 packets/second. The traffic pattern on the random network is many-to-one, and there are two sinks in the network such that half of the source nodes send to one sink, and the other half send to the other sink. Our random network results are an average of 50 simulations, each on a different 100 node random network, each of which has a different randomly chosen seed for both source positioning and data generation rate. For all other cases, results are an average of 25 simulations, each with a different randomly selected seed and each lasting for 4000 seconds.

Figure 5(a) shows data delivery capabilities of all four protocols on the 15-node chain network. RMAC's data delivery ratio starts deteriorating sharply when the packet rate exceeds 6 packets /50 seconds; this is due to RMAC's limited ability to adapt to higher traffic loads. DW-MAC maintains a higher delivery ratio compared to RMAC, however, on the average of the range of data rates covered, its delivery ratio is 73%. CL-MAC achieves a 25% improvement over DW-MAC which can be explained as follows. First, unlike DW-MAC, CL-MAC segment duration is traffic load dependent, and therefore nodes can reserve longer segments for transmission as needed. Second, within a segment, CL-MAC supports multiple packet transmission.

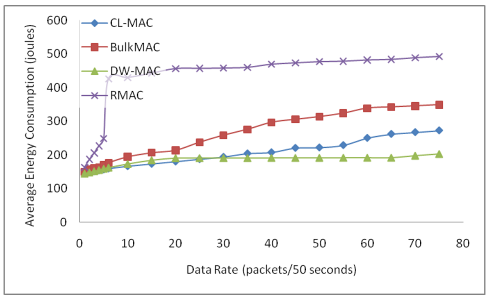
5(b) shows the average packet delay for the 15-node chain network. Compared to DW-MAC, average end-to-end delay for CL-MAC is 5% less and for RMAC, it's 250% more. Under low traffic, RMAC outperforms both DW-MAC and CL-MAC. CL-MAC's and DW-MAC's end to end delay is very close to each other until 6 packets/50 seconds. However, as the traffic rate increases, end-to-end delay for both DW-MAC and RMAC increases rapidly. CL-MAC outperforms DW-MAC and RMAC as it automatically reserves more segments when traffic load increases. We observe that after the data rate exceeds 27 packets/50 seconds, DW-MAC shows better results. DW-MAC's average delay jumps when the packet rate exceeds 12 packets/ 50 seconds, as it can no longer handle the traffic load. The reason for RMAC's



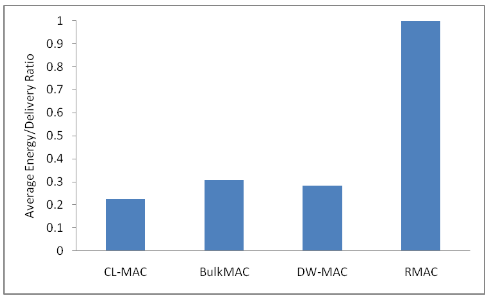
(a)



(b)



(c)



(d)

Figure 5. 15-node Chain Network: (a) Average Delivery Ratio, (b) Average Delay, (c) Average Energy Consumption, and (d) Average Energy Consumed/Packet Delivered

unstable performance at higher data rates is its extremely low delivery rate as shown in figure 5(a).

5(c) illustrates average energy consumption on the chain network. On average, DW-MAC consumes less energy compared to RMAC and CL-MAC. RMAC's high energy consumption can be explained by its high collision rate. CL-MAC's energy consumption increases directly proportional to the data rate, which can be explained by its on-demand channel reservation scheme discussed earlier in this section. DW-MAC energy consumption stays almost constant after packet rate exceeds 15 packets/50 seconds. Beyond this data rate, DW-MAC works at its full capacity and its flow scheduling policy minimizes packet collisions. The energy consumption gap between CL-MAC and DW-MAC is further explained below.

The average energy consumption results may be misleading if they are not correlated with delivery rate results, hence, in 5(d), we illustrate the ratio of the average energy consumption to the delivery rate for all four protocols. This is an indication of how much energy is spent to successfully deliver one packet. We can see that CL-MAC consumes 21% less energy per packet delivered when compared to DW-MAC. In addition, CL-MAC has the advantage of better end-to-end delay as shown in figure 5(b).

Figure 6(a) shows the data delivery ratio for the cross-chain scenario which captures how well the examined protocols perform in the presence of multiple flows. In this scenario, the node at the intersection of two flows creates a bottleneck. Source nodes generate data packets at a constant rate which is increased from 1 packet/50 seconds to 75 packets/50 seconds. When more than one packet is generated within a cycle time of DW-MAC or RMAC, around 9 packets/50 seconds, these protocols' performances start to deteriorate significantly. CL-MAC adapts to increases in traffic load better than DW-MAC, RMAC and BulkMAC. It maintains a delivery ratio close to 100% until the packet rate exceeds 45 packets/50 seconds. Furthermore, the standard deviation for the average delivery ratio of CL-MAC, BulkMAC, DW-MAC and RMAC is 13%, 21%, 33% and 37%. This shows CL-MAC's consistent performance and reliability. Averaged over the entire data rate range covered in the simulations, packet delivery ratios of CL-MAC, BulkMAC, DW-MAC and RMAC are 83%, 61%, 57%, and 48% respectively. Compared to the chain network, the performance gap between CL-MAC, BulkMAC and DW-

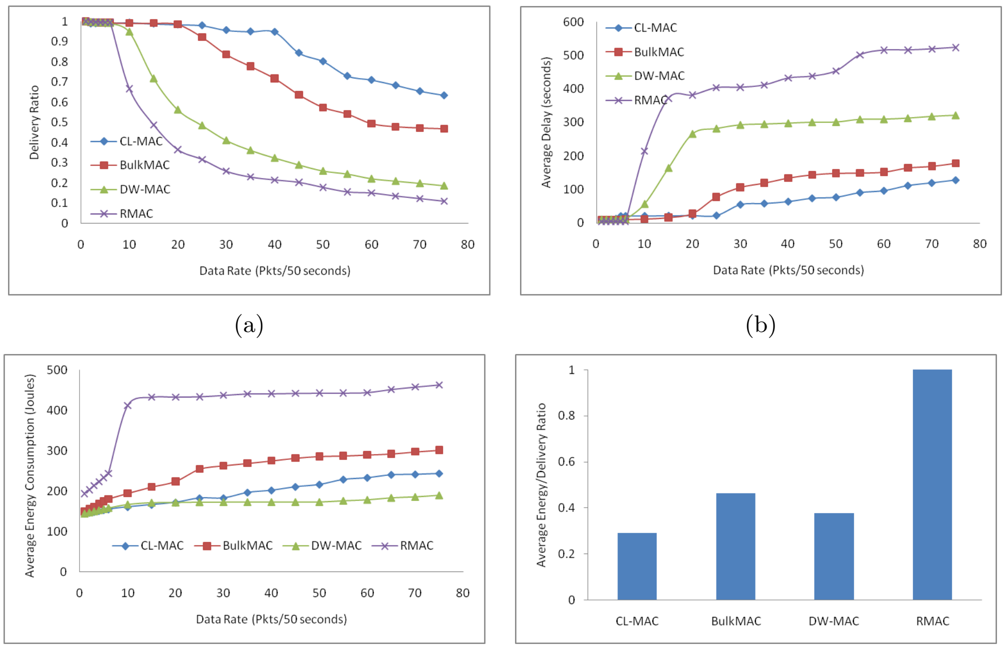


Figure 6. 25-node Cross Chain Network: (a) Average Delivery Ratio, (b) Average Delay, (c) Average Energy Consumption and (d) Average Energy Consumed/Packet Delivered

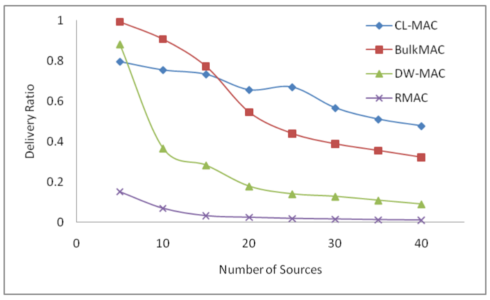
MAC, RMAC increases on the cross-chain network. This is due to inherent multi-flow support of CLMAC and BulkMAC, however, CL-MAC has the advantage of scheduling more packets over longer paths and the capability of simultaneously handling different data rates (i.e., generated from different sources as explained above).

6(b) shows the average end-to-end delay for the cross-chain network. Averaged over the entire data rate range covered in the simulations, the end-to -end delay for CL-MAC, Bulk-MAC, DW-MAC, and RMAC is 72.8 seconds, 180.5 seconds, 194.4 seconds and 289.8 seconds, respectively. When the traffic rate is low, RMAC outperforms all the other protocols. Bulk-

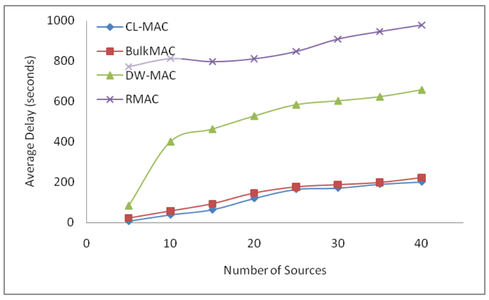
MAC's and DW-MAC's end-to-end delays are very close up to 6 packets/50 seconds. This is due to its simpler scheduling scheme and lower control overhead. End-to-end delay results on the cross-chain network reflect the effectiveness of multi-flow support in CL-MAC, which leads to a reduction of 60%, 63%, and 75% in average delay with respect to BulkMAC, DW-MAC, and RMAC, respectively.

The average energy consumption for the cross-chain network is illustrated in 6(c). On average, CL-MAC, BulkMAC and RMAC use 1.125, 1.35 and 2.18 times more energy than DW-MAC. RMAC's high energy consumption is due to its high collision rate as mentioned above. CL-MAC adapts to increases in traffic by reserving longer time segments for packet transmissions, hence it uses more energy. We again observe that CL-MAC's energy consumption increases as the traffic rate increases. This behavior can be explained by CL-MAC's traffic-adaptive flow-aware channel reservation scheme. As performed on the chain network, we measure the amount of energy consumed per packet delivered as shown in figure 6(d). The results are normalized to the largest value (i.e., RMAC).

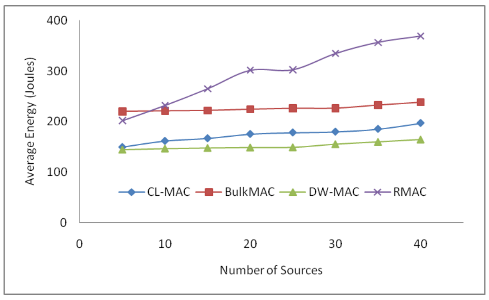
Figure 7(a) shows the delivery ratio of many-to-one traffic on the random network scenario. There are two sink nodes, half of the data sources send to one sink, while the other half sends to the other. Source nodes are randomly assigned to one of the sink nodes. This scenario examines the MAC protocols' ability to handle cross-traffic under data collection applications. The percentage of source nodes is increased from 5% to 45%. Averaged over the entire traffic generation range, CL-MAC, BulkMAC, DW-MAC and RMAC deliver 65%, 59%, 28%, and 5% of the generated data packets, respectively. RMAC's poor performance is due to packet collisions



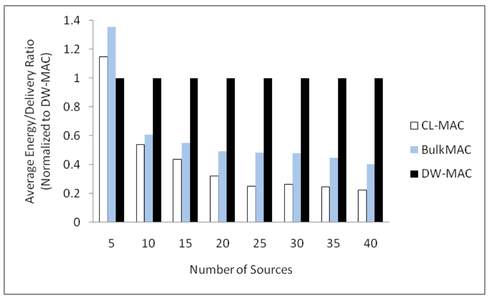
(a)



(b)



(c)



(d)

Figure 7. 100-node Random Network: (a) Average Delivery Ratio, (b) Average Delay, (c) Average Energy Consumption, and (d) Average Energy Consumed/Packet Delivered

resulting from RMAC's policy of scheduling data packet transmissions at the beginning of sleep period. In this scenario, multi-flow support is most needed due to the presence of multiple bottlenecks, i.e., intersecting flows. Compared to the other protocols, another advantage of CL-MAC that wasn't exposed in cross-chain and chain scenarios, is the ability of sink nodes to accept more requests from their neighbors as they can confirm flow setup requests in the sleep period via *EACK*.

Figure 7(b) shows end-to-end delay of many-to-one traffic on the random network. Averaged over the entire traffic generation range, CL-MAC, BulkMAC, DW-MAC and RMAC deliver a packet in 108 seconds, 140 seconds, 491 seconds, and 855 seconds, respectively. End- to-end delay illustrates how effectively CL-MAC adapts to traffic load variations.

Average energy consumption of many-to-one traffic on the random network is shown in figure 7(c). On average, when compared to DW-MAC, CL-MAC, BulkMAC and RMAC consume 14%, 48% and 84% more energy, respectively. CL-MAC and BulkMAC consume more energy because they adapt to increases in traffic load by scheduling more data packet transmissions. However, CL-MAC delivers more packets compared to all the other protocols which is illustrated in figure 7(d). Note that RMAC's energy consumption per packet delivered was significantly larger than the other three schemes, and was therefore omitted from the comparison in 7(d).

2.5 Summary

In this chapter we described CL-MAC, a Cross-Layer MAC protocol suitable for both homogeneous and heterogeneous WSNs. CL-MAC considers all pending packets in the routing buffer and pending flow setup requests (from other nodes) in its scheduling mechanism. This enables nodes to make more informed decisions when setting up flows. The proposed protocol is based on a unique structure of reservation packets, referred to as *FSPs*, which effectively utilize the sleep period. This gives a single node the capability of scheduling multiple multi-hop flows to multiple destinations, in a single cycle, leading to significant improvements in latency and throughput.

Through various rigorous simulations, we showed the effectiveness of CL-MAC in handling variable traffic loads and patterns. CL-MAC showed significant improvement, over state of the art, in latency and throughput while sacrificing negligible energy. In addition, CL-MAC's performance has shown consistency over a wide range of data rates and networks.

We believe that most future WSNs will incorporate heterogeneous sensors. This calls for network protocols that can efficiently handle varying traffic loads and patterns, from different sensors, to facilitate communication between heterogeneous nodes. Performing the classical task of balancing energy efficiency, end-to-end latency, and throughput in a single-field environment does not reflect conflicting demands a node suffers in a heterogeneous WSN. Therefore, considering heterogeneity of WSNs, is crucial to arriving at more realistic simulations.

CHAPTER 3

SUPPORTING MULTI-HOP AND MULTI-PACKET TRANSMISSION IN ASYNCHRONOUS WSNS

3.1 Introduction

Conventional network protocol design approaches, following the Open Systems Interconnection (OSI) layered architecture, regulate interactions between adjacent layers and set rigid functionality boundaries across layers. Although this helps standardize, simplify and accelerate the design process of network protocols, performance gains achieved are local (i.e., gains related to an individual layer) (8) (45). This does not guarantee an improvement in the overall network performance, which involves interaction with other layers (on the same node), other nodes, in addition to network and channel conditions.

As discussed above in section chapter 1, energy efficiency is desired in any communication system, however, in resource-restricted environments, such as WSNs, it becomes an ultimate priority. The tighter energy constraints in WSNs and the limitations of conventionally designed network protocols (i.e., following the OSI model) led researchers to explore CL design approaches. In CL approaches, possible gains from violating/modifying the OSI model, when designing network protocols, are explored (46) (47).

A variety of CL protocols for WSNs exists in the literature, varying from utilizing CL information (information from a different layer) in optimizing the operation of another layer to completely merging the functionality of several layers (8) (45). We focus on the former group, a more detailed discussion about different CL design approaches and paradigms is discussed in section 5.1. In this chapter, we specifically focus on CL interaction in asynchronous contention-based MAC protocols (see section 2.2 in chapter 2 for a brief classification of MAC protocols).

In asynchronous duty-cycled protocols, where nodes do not coordinate wakeup/sleep schedules, a few CL protocols exist, however, with a focus on routing. Some studies considered sharing MAC layer information at the routing layer to achieve least end-to-end latency routes such as (13) and (14). In (13), Kim et al. extended the solution of the routing problem solved in (48) to suit event-based/opportunistic routing in low duty-cycled sensor networks. In (14), the routing problem over time-varying transmission latency from one node to another was approximated and modeled as a time-dependent Bellman-Ford problem. However, neighborhood wakeup schedule information was assumed to be known.

Compared to synchronous contention-based MAC schemes, little attention has been given to CL interaction between routing and MAC layers to benefit from routing layer information at the MAC level. This is partly due to the limited expected benefit from such information if the node doesn't know when its neighbors wakeup. That is, routing information available at a node unaware of its neighbor's wakeup schedules, cannot be efficiently utilized. On the other hand, synchronization can be costly, especially when realized at a neighborhood level (i.e., all nodes within each others communication range coordinate wakeup and sleep).

In this chapter, we propose a CL design approach for asynchronous MAC schemes to support transmission of multiple-packets over multiple-hops and hence improve latency and throughput of asynchronous WSNs. The new approach utilizes routing information (multi-hop and multi-packet info) at the MAC level and communicates duty-cycling information to temporarily synchronize next hop nodes. We study the cost of sharing information within a node (between layers) as well as between nodes, which is reflected in extra memory requirements and communication overhead, respectively.

Since the gains resulting from information sharing are dependent on how a protocol utilizes such information, they cannot be isolated from the protocol behavior. This will vary amongst different protocols and according to network status variations. Therefore, in order to explore potential benefits of the proposed approach, it is necessary to use it in modifying current asynchronous MAC protocols.

We apply the proposed approach to modify the design of RI-MAC, a receiver initiated asynchronous MAC protocol for duty-cycled WSNs (49). The modification process follows the proposed framework, which gives RI-MAC unique awareness capabilities via communicating wakeup information (between nodes) in response to shared routing information (between layers). The redesigned RI-MAC (RI-MAC*) was implemented in NS-2 (50) and simulations were run for different network topologies, data rates and flows. We show that the proposed design outperforms RI-MAC in each simulated scenario. For example, on a 25-node cross-chain network, compared to RI-MAC, RI-MAC* reduces latency by 40% without increasing power consumption or compromising throughput.

3.2 Cross-layer Information: Representation and Communication

The information shared between layers within a single node and that communicated between different nodes, needs to be clearly quantified. We use the size of such information as a cost metric for both communication and storing. The granularity (resolution) at which the information is communicated dictates the communication overhead. We assume:

1. The cost of sending or storing one bit is unity (sending and communicating are used interchangeably in the text as well as storing and representing).
2. The information available to every node from other nodes is adaptable.
3. We define link cost to be a combination of transmission and queuing delays.
4. m is the total number of parameters (i.e., information of interest).
5. We divide CL information into *parameters*, which is also the term we use in reference to such information.

Assuming a total of m parameters (CL info.) of interest, $C_1, C_2, ., C_m$, each of which has a coefficient reflecting its importance (priority) to the run-time application. This assumption extends CL information sharing to include the application layer. We consider the entire spectrum of granularity in communicating and representing CL information. Communication and representation refer to inter-nodal information sharing and intra-nodal information sharing, respectively.

A node having k_x bits representing the parameter of interest C_x, of the total m parameters of interest, will end up storing $\sum_{x=1}^{m} k_x$ bits ($K.m$ if all parameters have the same resolution) to represent such parameters in a stand-alone fashion (no interaction occurs between those parameters). The coefficients of those parameters will also need $\sum_{x=1}^{m} l_x$ bits to be represented, where l_x is the number of bits representing the coefficient corresponding to the x^{th} parameter of interest ($l.m$ in case of all coefficients having the same resolution).

An example effort function given in (Equation 3.1) is assumed for illustration purposes, more details about this assumption can be found in (51). We assume that (Equation 3.1) reflects the effort needed from a node i to satisfy run-time application demands using current resources available to it. Run-time application demands are represented by coefficients $c_1, c_2, ..., c_m$ and priorities $P_1, P_2, ..., P_m$, which both have a directly proportional relationship with application demands. Current node resources are captured by the parameters, $C_1, C_2, ..., C_m$. The effort function is computed by each node and represents the minimum amount of information to be communicated between nodes.

$$E_i(c, C, P) = P_1 \frac{c_1}{C_1} + P_2 \frac{c_2}{C_2} + ... + P_m \frac{c_m}{C_m} \tag{3.1}$$

where P_m is the priority of coefficient c_m with to other coefficients and thus $\sum_{x=1}^{m} P_x = 1$. P_m is represented by q bits, and hence, a total of $q \times m$ bits is needed to represent all priorities individually.

The minimum number of bits (information) to be communicated will vary according to the resolution (number of bits) of parameters, their priorities and coefficients as well as the effort function formulation. For example, E_i, as shown in (Equation 3.1), has $m - 1$ additions, m divisions and multiplications. To be represented, each multiplication will require the sum of bits of both operands of the multiplication; we approximate division to have the same cost as multiplication. Addition requires the output to be as large as the larger operand of addition, i.e.,$max(t_1, t_2, , t_m)$, where t_1 is the number of bits the first term in E_i requires and t_2 is the number of bits representing the second term and so on. The x^{th} term of E_i will require t_x bits to be represented, and the total number of bits needed to represent E_i is $T_{min} = max(t_1, t_2, .., t_m)$ which will set the lower bound of information communication between nodes. Note that for every t_x there exists $q \times l_x \times k_x$ possible values.

On a finer level of granularity, each node may communicate each term of E_i individually rather than E_i as a whole. In such case, the total number of bits communicated will be increased to $T_{int} = \sum_{x=1}^{m} t_x$. The finest level of granularity will require communicating$T_{max} = \sum_{x=1}^{m} \sum_{y=1}^{n} N_x^y$, where N_x^y is the number of minimum size of bit-groupings needed to individually represent component y of term x in E_i. For example, in the above E_i given in (Equation 3.1), assuming the minimum size to represent any variable is one byte, even if each component of the x^{th} term of E_i requires less than one byte (e.g., 2 bits), the minimum value of N_x^y is one byte and hence each component of that term will require one byte to be represented (three bytes total). However, for generality, we consider the special case where the minimum size bit-grouping is one bit and therefore, $T_{max} = T_{int}$.

3.3 Routing and Medium Access

Like other CL protocols, we do not consider routing in isolation from medium access (13) and (14), however, the interaction is different. We consider adaptive duty-cycling as in (14), illustrated in Figure 8. Note that nodes do not follow a fixed wakeup schedule or duty cycle i.e.,

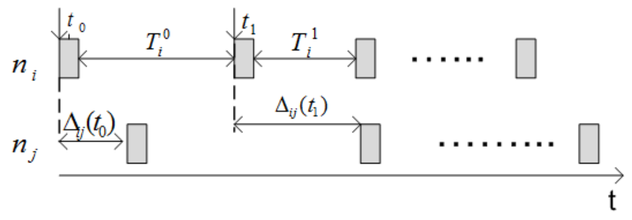


Figure 8. Randomly Adaptive Duty-cycled Network.

$T_i \neq T_j and T_i^0 \neq T_i^1$. We abstractly model the underlying MAC protocol to be the probability of the transmitter successfully accessing the channel while the receiver is awake, as shown in (Equation 3.2).

$$P_{ij}(t) = \frac{1}{N(t)} \times \frac{1}{\Delta_{ij}(t) + 1} \tag{3.2}$$

Where $P_{ij}(t)$ is the probability of a transmitter node i acquiring the channel to send a packet to a receiver node j, $N(t)$ is the number of neighbors contending for the channel at time t, and $\Delta_{ij}(t)$ is the difference in wakeup times between node i and node j at time t as shown in Figure 8. Note that in the synchronous case, $\Delta_{ij}(t)$ is known.

$P_{ij}(t)$ is mainly dependent on the number of neighboring nodes contending for the medium and their wakeup schedules. Since wakeup schedules may be synchronous or asynchronous, we consider the more general asynchronous case (randomly-varying duty-cycled networks). This scenario can be modeled as in (14), which attempts to solve the routing problem by predicting $\Delta_{ij}(t)$ which is considered unknown due to modularity of OSI-like design paradigms. We try to efficiently solve the routing/medium access problem as well, however, by following our new design approach.

3.4 Applying The Proposed Approach

In this section, we follow the above approach in modifying an existing asynchronous duty-cycled MAC protocol RI-MAC (49). We examine possible benefits of CL information sharing on both inter-nodal and intra-nodal levels. The new design of RI-MAC, referred to as RI-MAC*, leads to significant performance improvements as illustrated in section .

3.4.1 RI-MAC Overview

RI-MAC (49) is a receiver initiated asynchronous MAC protocol. Nodes wake up periodically, but each node wakes up independent of the other nodes and according to a random value (RV) between $0.5 \times L$ and $1.5 \times L$, where L is the sleep interval. Whenever a node wakes up and senses an idle channel, it broadcasts a beacon packet, which includes its address (similar to a preamble). If a node has a packet to send, it wakes up, and waits for its intended receiver's beacon. In preamble-based protocols, such as (32), (33) and (34), during the time when a node sends preambles, its neighbors cannot use the wireless medium. RI-MAC eliminates preambles, and therefore increases channel utilization.

Collisions in RI-MAC are resolved as follows: assume nodes A and C want to send data to node B which is within the communication range of both A and C. RI-MAC sets the initial contention window to 0 in the first beacon (sent by the receiver B), and hence, a collision occurs. Node B detects the collision and informs nodes A and C via another beacon with a non-zero contention window. Then nodes A and C send data based on the order determined via their back-off timers which were set in respond to the new non-zero contention window. Like other asynchronous MAC protocols, multi-hop packet transmissions will suffer large delays, and broadcast is more costly compared to synchronized MAC protocols.

3.4.2 RI-MAC*

The new design incorporates both dimensions of information sharing in RI-MAC's original functionality, namely intra-nodal and inter-nodal. The three main parameters of interest are:

1. latency reduction, which represents the application demands (intra-nodal).
2. node duty-cycling information, which is communicated among nodes (inter-nodal).
3. multi-hop and multi-packet information shared between routing and MAC layers of the same node (intra-nodal).

The first parameter of interest (referred to as coefficient in section II reflects application needs, the second parameter is used to minimize $\Delta_{ij}(t)$ in (Equation 3.2) rather than predicting it as in (14), and the third parameter is shared (between layers) to adjust the MAC behavior which will vary according to multi-hop and multi-packet traffic presence. Note that the second and third parameters of interest are node resources (can be mapped to the parameters C_1 and C_2 in (Equation 3.1)) used to achieve the application demands (reflected in the first parameter which can be mapped to the coefficient c_1 in (Equation 3.1)). The third parameter of interest is similar to other CL designs, which expose routing layer information at the MAC layer and factor it in the MAC decision (10) (12) (40). However, all those studies target synchronous schemes, or in other words lack the inter-nodal information communication dimension.

The two main CL parameters mentioned above (duty-cycling info and multi-hop and multi-packet info) give RI-MAC the ability to detect multi-hop and multi-packet traffic and temporarily synchronize senders and receivers in an on-demand fashion, accordingly. Detection of such traffic loads and patterns is achieved by factoring in the routing layer buffer in the MAC decision (CL info.). Temporary synchronization is achieved by adding a parameter field, containing RV, to the beacon, referred to as information beacon (I-beacon) in Figure 9(b). This allows the I-beacon receiver to compute the next wakeup time of the sender, and hence, wakeup synchronously until the last packet of the flow is received. The cost of adding RV to the beacon is its size (one byte in our simulations).

RI-MAC* is expected to outperform RI-MAC in multi-hop and multi-packet scenarios, however, it suffers from the extra cost of communicating duty-cycling information. This cost is expected to be easily offset in presence of relatively long paths between source and destination and presence of multi-packets to the same destination. To ensure that the savings always offset extra communication cost, we set a condition for temporary synchronization. The condition is: if the number of *packets* to a certain destination is larger than the number of *hops* between the source and destination. The logic behind such condition is that it will take at least n communication rounds to synchronize any two nodes n hops away from each other, this cost will not be offset unless there are more than n packets to be sent between these two nodes. This will reduce the large overall latency experienced by multiple packets and multi-hop traffic in adaptively duty-cycled WSNs.

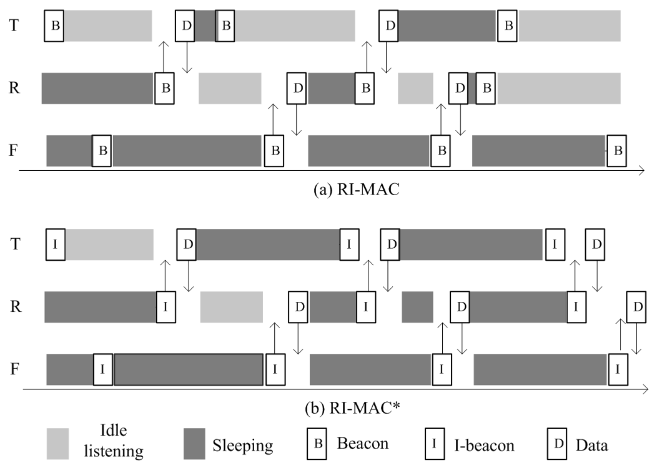


Figure 9. RI-MAC and RI-MAC* Flow Setup

Figure 9 illustrates the main differences between RI-MAC and its new CL version, RI-MAC*. Node T has n packets to send to node F through node R. Note that n is larger than 2 (the number of hops between T and F). In RI-MAC, T wakes up, senses the medium, and if sensed idle, it broadcasts a beacon and waits for possible data from neighbors and for R to wake up. R does the same as T, and since T has been waiting for R to wake up, it will send the first data packet for R immediately. R repeats what T did and waits for F to wake up, and so on. In RI-MAC*, node T initially broadcasts its *I-beacon* then waits for R to wakeup. Upon wake up, R sends its I-beacon which tells T when its next wakeup is (embedded in RV), however, T knows that R will change this time after its first communication with F. Upon the first communication with F, R computes its new wakeup according to the RV it received from F, however, R does not follow the new wake up time until it successfully notifies T via a new I-beacon. Finally, T, R and F will be waking up synchronously until the end of the multi-packet/multi-hop flow. We only focus on protocol-specific control and eliminate other control details for simplicity.

3.5 Performance Evaluation

We implement RI-MAC* using network simulator NS-2.29 (50), which was also used in (49). Each sensor node has a single omni-directional antenna and follows NS-2's combined free space and two-ray-ground reflection propagation model for WSNs. Table III summarizes the key parameters we used to simulate the radio of each sensor node. The parameters refer to the commonly used CC2420 radio, and use its RSSI sampling delay as the time for a single CCA (clear channel assessment) check as in (49). In NS-2, the transmission range and the carrier sensing range are modeled after the 914MHz Lucent WaveLAN DSSS radio. The MAC protocol parameters used are similar to those reported in (49) except for an extra byte used to communicate the parameter of interest (RV) as shown in Table IV. For simplicity, for both protocols, we assume no network partitioning and eliminate routing traffic, except for communicating the RV value in the I-beacon, which is the communication overhead in RI-MAC*.

TABLE III

SIMULATION PARAMETERS (RADIO)

Bandwidth	250 kbps	CCA Check Delay	128 μS
SIFS	192 μS	Interference Range	550 m
Slot Time	320 μS	Size of ACK	5B
Comm. Range	250m	Size of Hardware Preamble	6B

TABLE IV

SIMULATION PARAMETERS (MAC)

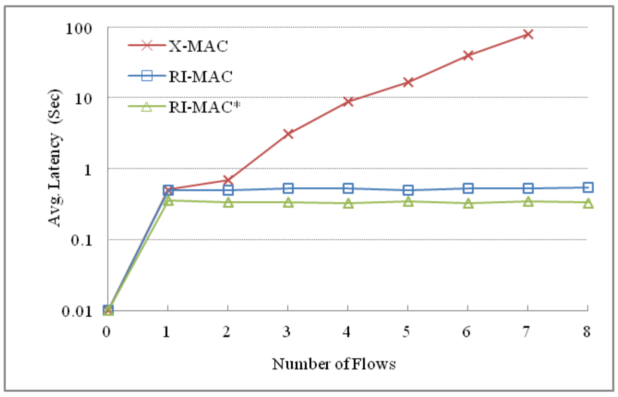
Protocol	X-MAC	RI-MAC	RI-MAC*
Backoff Window	32	0 - 255	0 - 255
Retry Limit	0 or 5	5	5
Special Frame	Short Preamble	Beacon	I-Beacon
Special Frame Size	6B	6-9 B	7-10 B
Dwell Time	10.5 ms	Variable	Variable

We compare RI-MAC's performance to that of RI-MAC* and X-MAC, under different traffic loads, patterns and network topologies. We define the three performance metrics of interest (a) Average latency, which is the time a packet takes to travel from source to destination (considers queuing delays on intermediate nodes), (b) Delivery ratio, the ratio of the number of packets delivered to their destinations to the total number of packets sent, and (c) Average duty cycle, the average ratio between wakeup time divided by the total of wakeup and sleep times. Note that the average duty cycle indicates energy consumption independent of the underlying hardware.

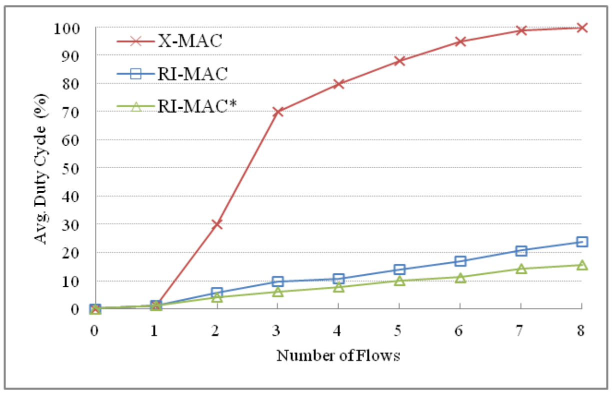
We test the performance of all three schemes on a 16-node clique network, a 10-node chain network, a 25-node cross-chain network, and on a 49 node (7x7) grid network. On the Clique network, all nodes are within the communication range of each other and the number of independent traffic flows is varied between one and eight, with no two flows sharing source or destination. Moreover, keeping the number of flows at eight, we vary the data rate from 1 to 8 packets/second. On the chain network, only one flow is generated by one end node and is transmitted across the network to the receiver at the other end. On the cross-chain, one flow from the end of each chain is generated and destined to the other end. All flows for the above three networks are of constant bit rate (CBR) and each simulation is an average of 20 random runs and lasts for 2000 seconds. A binary exponential backoff (BEB) is followed after each retransmission (52).

For the 49-node grid network, each node is 200 meters apart from its neighbors and the sink is placed at the center of the network (49). We used the Random Correlated-Event (RCE) traffic model (11) which simulates the impulse traffic triggered by spatially correlated events. RCE selects a random location $(x, y$ for each event. The only nodes that generate data to report the event are those falling within a virtual circle centered at $(x, y$ and with radius R. As in (49), we adjusted the sensor range R to simulate different workloads in the network. Note that the sensing range R also represents the radius of the virtual circle centered at $(x, y$ which specifies the circle of influence of an event. A new event is generated every minute and each node sensing the event reports that to the sink node via a single packet. R is varied from 100 meters to 500 meters and the average number of packets generated per event is reported in Table V. The average length of the path each packet traverses to reach the sink is 3 hops. Unicast packets are sent towards the sink node during every simulation run. The traffic is triggered by a series of 100 events and each average value is based on 30 random runs which allowed the confidence intervals to reach around 99%.

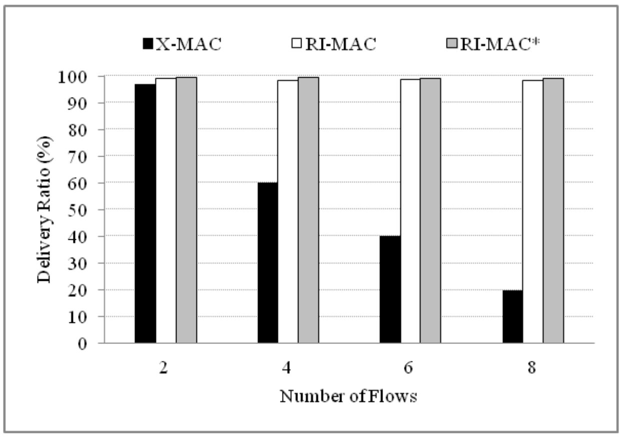
On the clique network, traffic is generated at a rate of 1 packet/second while varying the number of flows as shown in Figure 10. Compared to RI-MAC, RI-MAC* reduces average la-



(a) Average Latency



(b) Average Duty Cycle



(c) Delivery Ratio

Figure 10. 16-node clique network at a constant data rate of 1 packet/second

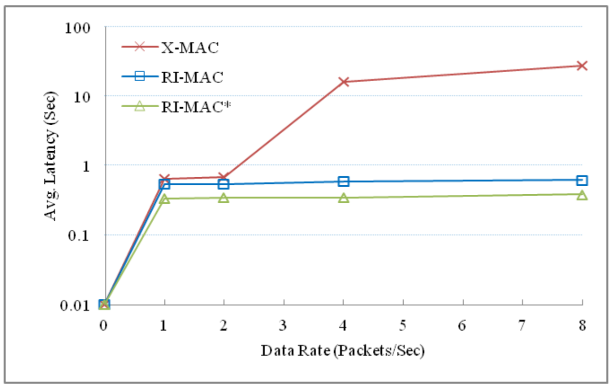
TABLE V

AVERAGE NUMBER OF PACKETS GENERATED PER EVENT FOR DIFFERENT SENSING RANGES

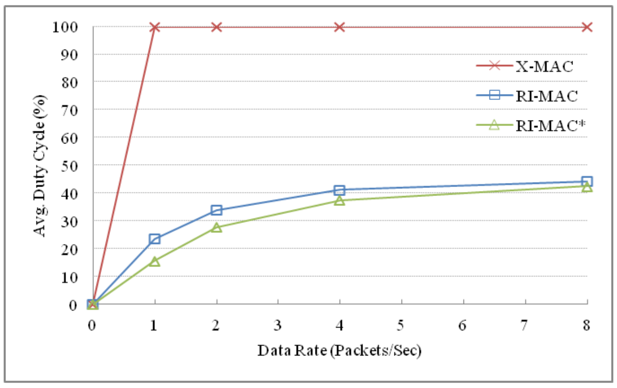
Sensing Range (m)	Generated Packets
100	0.9
200	3.2
300	6.8
400	10.8
500	15.6

tency, consumes less energy, while sustaining a delivery ratio close to 99% as shown in Figure 10 (a), (b) and (c), respectively. Although these are single-hop flows, they all consist of multiple packets, which activates temporary synchronization. Note that the communication overhead of I-beacons over regular beacons is inherently accounted for since they are larger by the extra byte each RV value occupies. Overall, RI-MAC* reduces the average latency and average duty cycle of RI-MAC by 34% and 31.8%, respectively. This is achieved with no penalty in delivery ratio. On the other hand, X-MAC is only capable of maintaining reliable throughput when the number of flows is 2 or below. The steep decline in delivery ratio is due to the preamble transmissions which saturate the network, and therefore increasing the number of queued packets.

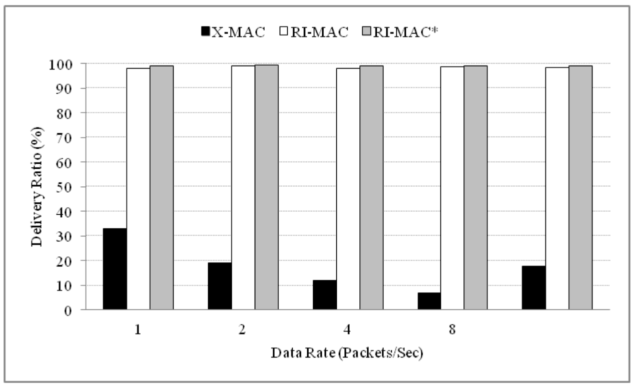
Figure 11 illustrates the effect of increasing the data rate while keeping the number of flows fixed on the 16-node clique network in order to test the capability of each protocol in handling higher traffic loads. We fix the number of flows at 8 (all nodes are involved in communication)



(a) Average Latency

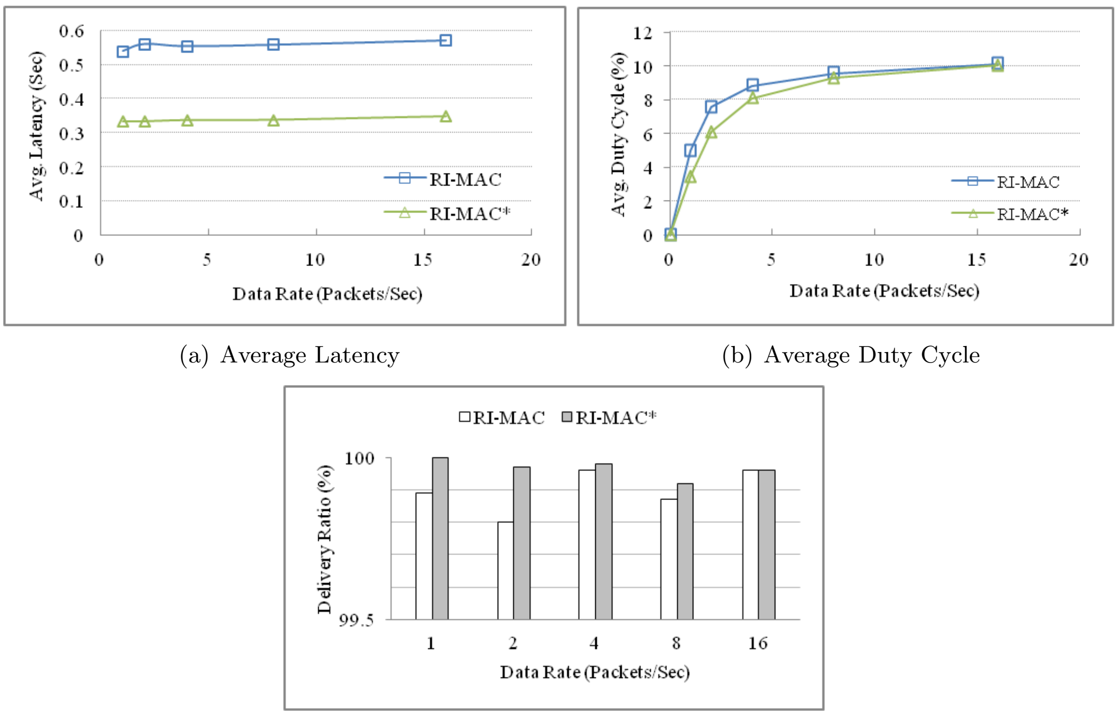


(b) Average Duty Cycle



(c) Delivery ratio

Figure 11. 16-node clique network with 8 flows and vary the data rate from 1 to 8 packets/second. Although RI-MAC* outperforms RI-MAC in this scenario as well, it is noticeable that the performance gap decreases as the traffic rate increases. This is due to I-beacon collisions which RI-MAC* is more prone to compared to RI-MAC. This is a result of its larger beacon size (I-beacon) which is, on average, 12.7% larger than the regular RI-MAC beacon. Moreover, it is evident that X-MAC can not reliably handle this type of traffic load due to its preambles saturating the medium.



(a) Average Latency (b) Average Duty Cycle

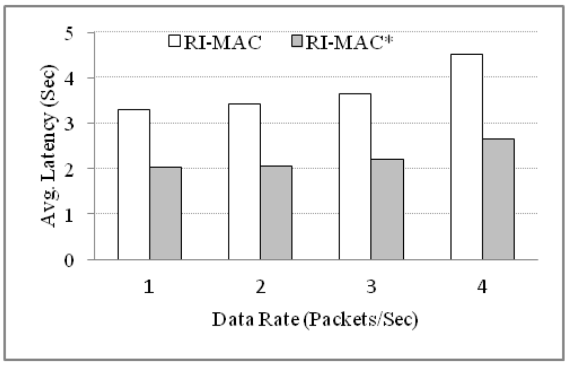
(c) Delivery ratio

Figure 12. 10-node chain network

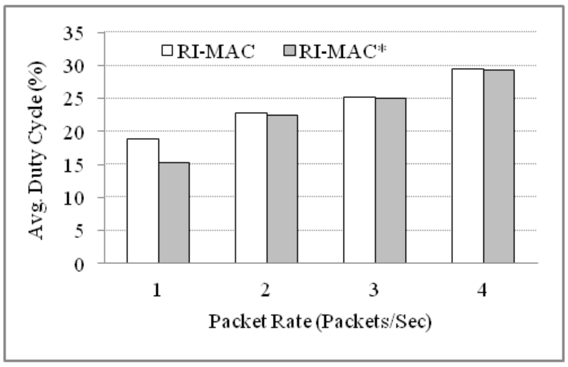
On the chain network, we increase the traffic rate from 1 to 16 packets/second, as shown in Figure 12. RI-MAC* reduces the average latency and average duty cycle compared to RI-MAC by 39.8% and 10.2%, respectively. A similar trend in the average duty-cycle difference is noticed, as in Figure 12, and can be explained in part by the above reasoning. However, there is an additional factor in this case. As the number of hops increases, nodes along the route will attempt to wake up in a staggered manner (closer to each other), however, there is no accurate synchronization reference. In other words, they are relatively synchronized to one another rather than to a single reference as in synchronous contention-based techniques. Such tightly synchronized techniques are expected to have smaller duty cycles, however, at the cost of complexity and throughput.

On the cross-chain network, RI-MAC* reduces latency by 40% compared to RI-MAC (averaged over all packet rates) as shown in Figure 13(a). This is achieved without consuming any extra power, as shown in Figure 13(b), and with a slight improvement in delivery ratio, as shown in Figure 13(c). Note that the duty cycle is an indication of power consumption i.e., the larger the duty cycle, the more power is consumed due to the longer wakeup time and vice versa. Note that X-MAC results were omitted from Figure 12 and Figure 13 due to very poor performance compared to RI-MAC and RI-MAC*.

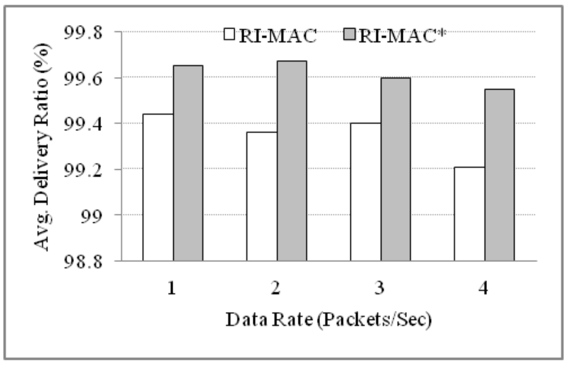
The performance of the three protocols are on the 49-node grid network are shown in Figure 14. Figure 14(a) shows the average and maximum latency of packets as the traffic load increases (i.e., sensing range increases). Both RI-MAC and RI-MAC* outperform X-MAC and that is mainly due to RI-MAC's clever scheduling which increases the medium idle time. RI-



(a) Average Latency



(b) Average Duty Cycle

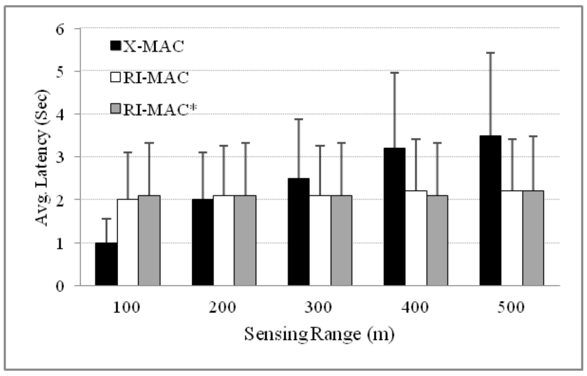


(c) Delivery ratio

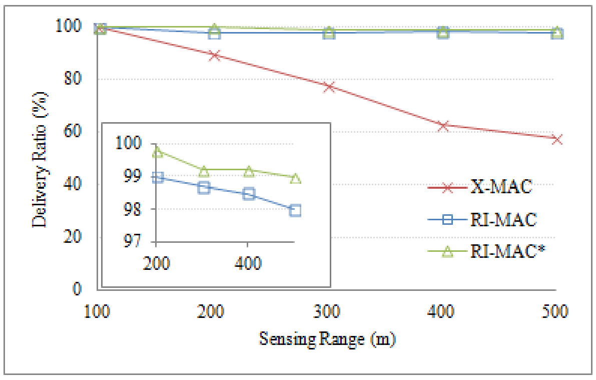
Figure 13. 25-node cross-chain network

MAC* slightly improves latency, however by a small margin. Throughput is not compromised and delivery ratio is maintained near 100% for both RI-MAC and RI-MAC*. However, Ri-MAC* has a modest advantage. The modest improvement in both latency and duty cycle is due to the random nature of event driven traffic patterns which does not guarantee the number of hops to be sufficiently large in order for latency improvement to offset the extra scheduling overhead. This is also reflected in the duty cycle results shown in Figure 14(c).

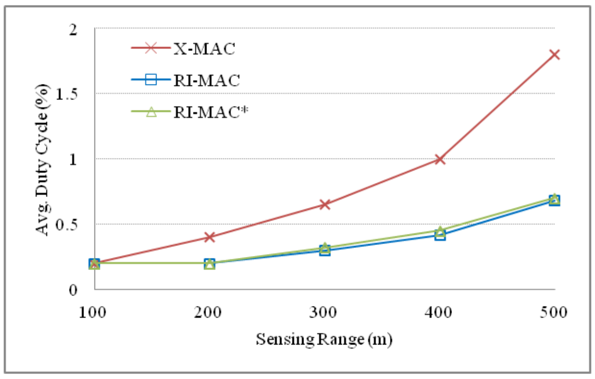
Although collisions cause more power consumption (reflected in larger duty cycles), the delivery ratio is sustained at almost 100% as shown in Figure 10(c), Figure 11(c), Figure 12(c)



(a) Average and maximum latency versus sensing range



(b) Delivery ratio versus sensing range

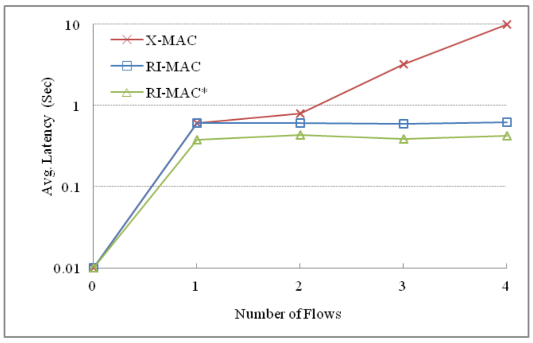


(c) Average duty cycle versus sensing range

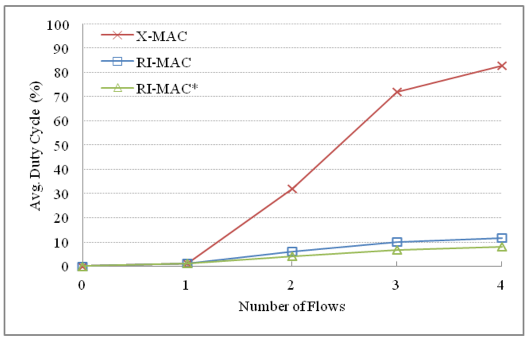
Figure 14. Performance for unicast traffic in the 49-node (7X7) grid network.

and Figure 13(c). RIMAC* does not trade off throughput for latency or power savings (33) (34), however, it can be simply done by reducing the number of retransmission attempts.

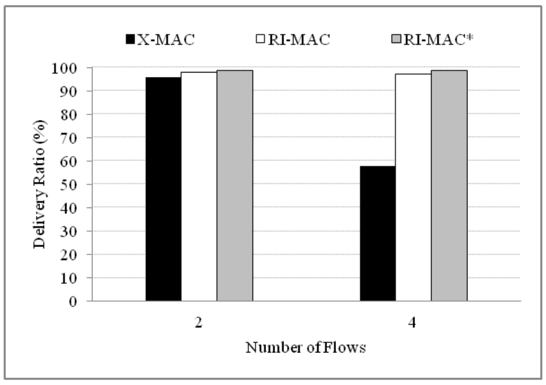
In order to validate our simulation-based evaluation reported above, and to explore hardware platform-dependent issues, we compared RI-MAC* with RI-MAC and X-MAC in a TinyOS implementation on TelosB motes. We implemented RI-MAC* for the CC2420 radio under the UPMA framework in TinyOS. The CC2420 is a packetizing radio used in TelosB and MICAz



(a) Average Latency



(b) Average Duty Cycle



(c) Delivery Ratio

Figure 15. 8-node clique network at a constant data rate of 1 packet/second (TinyOS)

motes, however, the code can be ported to motes with streaming radios. This follows the original RI-MAC TinyOS implementation described in (49). To account for software processing delays on the TelosB motes, we adjusted some parameters of RI-MAC*. A mote may suffer some delays before transmitting consecutive packets in the queue, such as post-processing of a transmitted packet, moving a queued packet to the MAC layer, and loading the packet to the hardware buffer. Therefore, like in RI-MAC implementation, we added an extra 10 ms to the dwell time defined in RI-MAC* to account for these delays. If a beacon were processed in hardware, this time could be much shorter.

In order to verify our simulations, we experiment RI-MAC* on TelosB motes in clique networks. These experiments are intended to replicate the simulation experiments we performed for clique networks, discussed above. The network configurations and traffic model are the same as we used in simulations, and each data point is an average of ten experiments each lasting for five minutes. The results are shown in Figure 15 and closely match our simulation results shown earlier in Figure 10. However, in our TinyOS experiments, the number of flows is limited to four as we only had eight TelosB motes available. We also observed that changing node placement caused performance variations by up to 10%.

3.6 Summary

This chapter presented a CL collaborative approach to support multi-hop and multi-packet traffic in asynchronous WSNs. The proposed approach realizes CL coordination via cross-layer information exchange and inter-nodal information exchange. The cost of sharing and communicating information is quantified along with potential benefits. To assess possible gains from adopting the new design approach, we modify RI-MAC (49) by integrating routing protocol functionality with that of MAC in addition to information sharing between nodes.

CHAPTER 4

A CROSS-LAYER APPROACH FOR CONTEXT-AWARE DISTRIBUTED DATA GATHERING APPLICATIONS

4.1 Introduction

Many data gathering and monitoring applications require nodes to continuously report sensory data, such as temperature and humidity, to an observer (e.g., base station or control center) in order to detect the occurrence of an event of interest or an anomaly. In these applications, it is common that the observer requires an aggregate value of the sensed field rather than individual values from each sensor. For example, in a traffic monitoring application where noise levels are monitored, the observer may be interested in the average value (24). In an industrial application monitoring the temperature of hazardous materials, the maximum measured value (or significantly different values) from all sensors in a neighborhood might be of interest in order to detect an emergency or anomaly. The salient characteristic of such applications is that they do not require data from each individual sensing node all the time, instead they may require receiving all unique or ”representative” data values with a certain degree of tolerance set by the application. We refer to such operations as context-aware data gathering.

Most existing data gathering solutions in WSNs employ some type of clustering technique to facilitate data collection (53). Such techniques divide the network into a number of groups called clusters. Each cluster has a representative node referred to as the Cluster Head (CH), and a number of member nodes. Conventionally, the CH receives data from each member node, then processes/aggregates this data before sending it to the observer, directly or via intermediate CHs. This data gathering, between the cluster members and the CH, does not consider the context for which the data is collected and results in redundant communication operations. For instance, consider the industrial monitoring application above, assume that each cluster consists of N nodes. If all nodes within a cluster are sensing the same value (or values within a certain threshold specified by the context), in conventional clustering/data gathering approaches, the CH will end up processing/aggregating N similar values, $N - 1$ of which are redundant.

For simplicity, existing data gathering techniques either ignore peculiarities of the WSN Medium Access Control (MAC) layer or assume contention-free channel access in intra-cluster and inter-cluster communications (53) (54) (55). For example, HEED (55) makes no assumptions about MAC. On the other hand, in LEACH (54), after cluster formation, nodes access the channel following TDMA schedules assigned by the cluster head, and inter-cluster communications follow different CDMA codes, which are both contention-free schemes. However, most contention-free MAC schemes suffer from strict synchronization requirements and poor adaptability to changes in network topology which results in poor scalability (8) (53). For example, when a new node joins a cluster in a TDMA MAC scheme, slot re-assignment is required along with updating the frame length, which are non-trivial tasks.

Although prolonging the WSN lifetime is a critical matter, as stated above, other attributes such as scalability and adaptability to changes in network size and topology, and node density should not be compromised. This motivated exploring contention-based MAC schemes in WSNs (26). Despite their adaptability and scalability, these schemes are more prone to major sources of energy waste, compared to contention-free schemes. Namely, overhearing, idle listening, collisions and control packet overhead. Overhearing occurs when a node receives a packet not destined to it, idle listening occurs when a node listens to a traffic-free channel, and collisions cause reception of corrupted packets. Different techniques have been proposed to address these sources of energy waste. For example, periodically alternating the radio between ON and OFF states (duty cycling) to reduce energy waste from idle listening and overhearing (26).

We propose a Cross-Layer (CL) solution to reduce communication redundancy, balance the load, and hence prolong network lifetime in data gathering/monitoring applications in WSNs. The proposed solution mainly targets applications that are interested in representative data values from each geographical region (e.g., event/anomaly detection problems) and does not aim to reconstruct the entire sensing field. We present a distributed data gathering technique that realizes Dynamic Virtual Clusters (DVCs), called DVC-DG (Dynamic Virtual Clustering-based Data Gathering). DVCs are formed via an Overhearing-Dependent MAC scheme (OD-MAC). During every communication round, nodes compete (contend) for channel access and the winner of the channel transmits data to the sink and becomes the representative node (CH) of all the nodes having a similar data value within its communication range. The CH and member nodes within its communication range form a data-dependent virtual cluster. Cluster members overhear the channel and compare the data being communicated by the CH over the channel to their own data. If found similar (within a given threshold dependent on the context), they suppress their transmissions. Therefore, data redundancy is eliminated at its very source and not at the CH. This approach reduces the number of intra-cluster communications and collisions significantly, hence saving energy.

To the best of our knowledge, no other data gathering solution integrates overhearing to form DVCs in its functionality in order to improve energy efficiency and prolong network lifetime. We analytically demonstrate the energy efficiency of the proposed approach. Despite the cost of additional overhearing, our results show a reduction in the number of communication operations by a factor of up to *N-1*, where N is the number of nodes in a neighborhood. We also study the cost of overhearing in terms of neighborhood degree and present the associated energy analysis.

The remainder of this chapter is organized as follows: Section 4.2 is a literature review of relevant studies from the clustering and MAC domains. In Section 4.3, we formulate our problem. In Section 4.4, we discuss the proposed data gathering technique and the formation of DVCs, followed by an analysis of its underlying MAC scheme in Section 4.5. We evaluate the proposed technique in Section 4.6, followed by the conclusion in Section 4.7.

4.2 Related Work

Many data gathering and monitoring applications benefit from spatial correlations in the monitored field to improve communication efficiency, hence saving energy. Exploiting spatial correlation of the sensed field at the MAC layer in WSNs was investigated in several studies (56) (57). Varun et al. proposed a Correlation-based Collaborative MAC (CC-MAC) protocol (56). CC-MAC is a Carrier Sense Multiple Access with Collision Avoidance (CSMA/CA) based MAC protocol that aims to prevent redundant transmission from neighboring sensors by using an iterative node selection algorithm which finds the number and location of representative (transmitting) nodes based on the vector quantization method.The statistical properties of the WSN topology are fed to the node selection algorithm, based on which it creates a sample topology of the network and the number of representative nodes is decreased iteratively to meet a preset distortion constraint.

Jamieson et al. proposed Sift, a CSMA-based protocol tailored for spatially-correlated contention, where multiple nodes in a neighborhood sense an event and contend for transmitting the sensed information (57). Only R out of N nodes in the neighborhood get to transmit their data, the remaining N-R nodes will suppress their transmission, this is done via a non-uniform probability distribution within a *fixed-size* contention window to pick up a slot. A high probability of success for channel access is achieved in addition to reduced collision probability, hence, achieving good throughput under variable traffic. However, the optimal probability distribution works only when senders always have data to transmit and they are synchronized for channel access. Thus, when data arrivals to a node are highly random and senders cannot sense each other for data transmission, its performance degrades to the case of CSMA with the uniform access probability distribution. This limits its applicability in energy-efficient MAC schemes which employ duty-cycling, as synchronization is required periodically.

The idea of utilizing overhearing in data gathering applications is not new, but we take the initiative in combining it with clustering solutions for data gathering applications. In (58), Time Division Multiple Access (TDMA) is adopted for MAC and nodes wakeup during their neighbors' assigned slots to artificially overhear their transmissions and hence determine spatial information redundancy. Nodes also apply a simple interpolation of their previous and current readings to determine temporal redundancy. In (59), overhearing is utilized to reduce control overhead. It is generated by having nodes wakeup at the end of each successful transmission to send/receive data without transmitting the long preamble, instead, Request To Send (RTS) and Clear To Send (CTS) messages are used. The idea of waking up during neighbor's communication was also proposed in (58) and (60).

In addition to employing an energy efficient MAC protocol, many data gathering/monitoring applications in WSNs employ a clustering technique to organize the data gathering and aggregation process (53). Clustering has been widely studied in the data processing and wired network domains. However, clustering schemes developed in these areas do not consider typical attributes of WSN deployment and operation. This limits their applicability to WSNs. Specifically, the ad-hoc fashion in which WSNs are deployed and the large number of location-unaware nodes forming the network suggests that distributed clustering techniques are more suitable. These techniques rely on neighborhood information rather than the entire network (i.e., do not assume a centralized controller aware of the network topology) (53). Moreover, energy efficiency and load balancing are crucial for prolonging WSN lifetime. The former can be addressed by reducing communication operations and control overhead, and the latter can be achieved by re-clustering. Finally, mobility and harsh environments may result in dynamic node distribution and unexpected failures. These can also be addressed via periodical re-clustering. In response to the above observations, several distributed clustering approaches have been proposed and can be broadly classified according to their execution as iterative or probabilistic.

In iterative clustering techniques, a node waits for other nodes' decisions or for a specific event occurrence before deciding its role (61) (62) (63). For example, in the Distributed Clustering Algorithm (DCA) (63), a node waits for all its neighbors with higher weights to decide whether to be cluster heads or join other clusters, before making a decision. The Weighted Clustering Algorithm (WCA) combines several factors in the weight metric used in clustering (64). Using a spanning tree to generate clusters with specific properties was proposed in (62). The main problem in most iterative techniques is the dependence of their time complexity (convergence speed) on the network diameter and the high sensitivity to packet loss (ripple effect).

Probabilistic clustering techniques allow each node to decide its role in the network independently while ensuring rapid convergence (i.e., terminate in constant time). LEACH is an application-specific data dissemination protocol which utilizes clustering to prolong network lifetime (54). It assigns a fixed probability to each node for electing itself as a cluster head which randomizes the process and achieves load balancing. The clustering process involves one iteration (like DVC-DG), after which a node decides whether to become a cluster head. However, each cluster head in LEACH broadcasts its decision (not required in DVC) and the number of cluster heads in the network is assumed to be known a priori. LEACH also assumes uniform energy consumption for cluster heads. Unlike LEACH, HEED (55) makes no assumptions about energy consumptions (like DVC) and selects cluster heads during protocol execution. Note that both (54) and (55) assume TDMA schedules in intra-cluster communications, which can limit scalability, adaptability and applicability to WSNs employing a contention-based MAC scheme.

In (58), TDMA is adopted for MAC and nodes wakeup during their neighbors' assigned slots to artificially overhear their transmissions and hence determine spatial information redundancy. Nodes also apply a simple interpolation of their previous and current readings to determine temporal redundancy. In (59), overhearing is utilized to reduce control overhead. It is generated by having nodes wakeup at the end of each successful transmission to send/receive data without transmitting the long preamble, instead, Request To Send (RTS) and Clear To Send (CTS) messages are used. The idea of waking up during neighbor's communication was also proposed in (60) (58).

LEACH (54) and HEED (55) are among the most popular data gathering protocols that integrate clustering and Medium Access Control (MAC) in their operation. LEACH is a data dissemination protocol which utilizes clustering to prolong network lifetime. It assigns a fixed probability to each node for electing itself as a CH which randomizes the process and achieves load balancing. The clustering process involves one iteration, after which a node decides whether to become a CH. However, each CH broadcasts its decision, and the number of CHs in the network is assumed to be known a priori. Unlike LEACH, which assumes uniform energy consumption for CHs, HEED makes no assumptions about energy consumption and selects CHs during protocol execution.

Although all the above techniques do a good job in achieving the common goal of reducing in-network traffic and/or improving latency, they are not optimized for applications that do not require field reconstruction or those surveilling random fields. We have the following observations:

- In (56), (58) and (59), there is no distinction between different types of data gathering applications. Hence, they attempt to reconstruct/estimate the entire underlying sensed field regardless of the data being sensed. This may lead to significant communication and data redundancy in certain data gathering applications such as anomaly detection.
- In (57), medium access priority is granted to the first R nodes which are assumed (a priori) to represent the entire N nodes in a neighborhood (i.e., there is no real time data processing). This can significantly limit its adaptability to field variations.
- In (54) and (58), nodes follow TDMA schedules (i.e., contention-free channel access). TDMA-like schemes require tight synchronization, which is not feasible in resource-constrained environments, such as WSNs, in addition to their implementation complexity and poor scalability.
- (56) and (58) assume knowledge of node location which might not be feasible in WSN environments, especially in presence of mobility or highly dynamic phenomena.
- (56) and (58) run an estimation scheme on each node based on the field correlation. This can be expensive from a computation/storage perspective as the WSN grows and can not be directly applied to random fields (i.e., uncorrelated fields).
- (56) iteratively elects the cluster head which can induce significant delays, especially in more dense and mobile WSNs.
- Overhearing utilization in (59) is limited to control packets and is only applied to the case of long-preamble Low Power Listening (LPL) MAC protocols. This is only true for LPL

protocols with large preambles such as B-MAC (27). However, many improvements over B-MAC that greatly reduce preamble size exist in the literature (33) (34).

In addition to the above limitations, most data gathering solutions do not integrate the design of MAC and clustering protocols. While this simplifies the design of each protocol independently, it may also lead to overseeing possible gains from combining the two processes (57). As a result, most clustering schemes require complex cluster head elections and re-elections to achieve load balancing (65) (66). A message complexity of $O(I.N)$ is required for each cluster head election process, where I is the number of iterations required before election termination and N is the number of nodes per cluster (neighborhood). Such message complexity may not be significant in small clusters/networks. However, as the cluster/network size and re-election frequency increase, it induces significant overheads (even if I is constant as in (54) and (55)).

4.3 Problem Formulation and Assumptions

Consider a set of N sensor nodes and a sink node attempting to collaboratively solve a binary hypothesis testing problem, where each of the N nodes is required to decide between transmitting its local sensed data (hypothesis H_1) to the sink node or not (hypothesis H_0). We assume that time is divided into discrete slots of equal durations, τ_s, in which a node can transmit/receive data. We also assume that nodes are within each others' communication range and listen to each others' transmissions (i.e., each transmission is a broadcast). The term slot and observation interval are used interchangeably.

Let $O_i(t)$ be the observed value at node i during slot t, and $S_i(t)$ be the sensed value at node i during the same slot, where $i = 1, 2, ..., N$. Notice that the observed value at node i is that transmitted by any of the other $N-1$ nodes in the network, while the sensed value is that sensed by node i itself. Elements constructing the sets of observed and sensed values at node i over a time span of T slots, $\{O_i(t)\}_1^T$ and $\{S_i(t)\}_1^T$, respectively, are independent given each hypothesis and are assumed to be identically distributed. $O_i(t)$ can be represented by:

$$\begin{aligned} &H_0 : S_i(t) - tol_i(t) + W_i(t) \leq O_i(t) \leq S_i(t) + tol_i(t) + W_i(t), \\ &\qquad\qquad\qquad\qquad\qquad\qquad t = 1, 2, ..., T \\ &H_1 : otherwise \end{aligned} \tag{4.1}$$

where tol_i is the permissible tolerance between the observed and sensed values, which reflects the level of accuracy required by the application. W_i is additive white Gaussian noise with a power of σ^2, assumed to be similar at all nodes. Following the analysis in (67) and (68), we arrive at:

$$Y_i = \sum_{t=1}^{T} log \left[\frac{f_{O_i(t)}(o_t|H_1)}{f_{O_i(t)}(o_t|H_0)} \right] \tag{4.2}$$

where $f_{O_i(t)}(o_t|H_1)$ and $f_{O_i(t)}(o_t|H_0)$ are the conditional probability distributions of $O_i(t)$ given H_1 and H_0, respectively. Y_i is the Log-Likelihood Ratio (LLR), computed by node i (see (69) for details). The signal-to-noise ratio (SNR) can be defined as $\psi_i = \sigma_{o_i}^2/\sigma^2$, where $\sigma_{o_i}^2$ is the average observed signal power at node i. This will result in an LLR computed at node i as follows (68):

$$Y_i = \frac{\psi_i}{2\sigma^2 + 2\psi_i\sigma^2} \sum_{t=1}^{T} |O_i(t)|^2 - log(1+\psi_i)\frac{T}{2} \tag{4.3}$$

Both (67) and (68) propose approximations for the above likelihood functions of Y_i given either H_0 or H_1, which are shifted scaled chi-square distributions with N degrees of freedom. Since the signal processing dimension of this work, where each node assesses observed signals (from other nodes) in order to evaluate the importance of its locally sensed data, can be solved similar to (68). We focus on the information processing dimension (i.e., spatial correlation) in the following sections.

4.4 The DVC-DG Technique

In this section we present the details the proposed Dynamic Virtual Clustering Data Gathering technique (DVC-DG). We shed light on the gains of exploiting overhearing in the clustering domain with a focus on data gathering and monitoring applications.

4.4.1 Overview

Dynamic Virtual Clustering-based Data Gathering (DVC-DG) utilizes CL collaboration between MAC and application layers to reduce communication redundancy. It employs overhearing to realize dynamic data-dependent virtual clusters (DVCs), and distributes the data aggregation responsibility over all cluster members. Every communication round, the node winning the channel sends its data to the sink while all other nodes within its vicinity overhear the ongoing transmission. If the overheard data is similar to what a node has, it suppresses its transmission, otherwise, it contends for the channel again to send its own value. The main unique characteristics of the proposed technique can be summarized as follows:

- The CH is the node winning the channel contention, which is inherently elected by the underlying MAC scheme. This eliminates the need for a special CH election algorithm.

- The CH does not communicate directly with cluster members, it only transmits its sensed data to the sink (base station).
- It is the CH's neighbors' responsibility to identify whether the CH represents them, based on overhearing the CH's communication to the sink. If the CH indeed represents them, they consider themselves cluster members.
- Nodes already represented by a CH (i.e., cluster members) suppress their pending data transmission. This eliminates data redundancy at its source and can significantly reduce congestions and collisions.

Similar to other distributed clustering schemes (e.g., (54) (55)), DVC neither assumes knowledge of nodes' location or network topology. However, in contrast, DVC does not involve any centralized data aggregation/compression at the cluster head nor performs field estimation based on observed field correlations as in (56) (58). This makes it applicable to virtually all data gathering and monitoring applications in WSNs. This also makes it suitable for applications involving random fields, where field correlation is not present. For example, in a surveillance application, where the presence of a specific object is monitored, sensors will either report a true or false type of data. In this scenario, the distribution of objects being tracked and their motion, which can both be random, are responsible for any similarity in reported data. Therefore, the field correlation assumption, based on which some protocols estimate the field, does not hold.

4.4.2 Assumptions

Assume the time needed to complete network clustering to be T_C, and the time between two successive clustering operations to be T_S. Due to high clustering cost, conventional clustering schemes require $T_S >> T_C$ to reduce overhead. This is not required in DVC as it distributes cluster head responsibilities over the entire cluster, and cluster head election inherently occurs via the underlying MAC scheme as discussed in section 4.5. We make the following assumptions about the network:

- All nodes are similar in terms of hardware/software capabilities, thus any node can act as a cluster head or a member.
- Nodes are deployed in an ad-hoc fashion on a circular field of radius r.
- Nodes are not equipped with a GPS and are unattended.
- Inter-cluster communications can either occur via long range communications (cluster head to cluster head) or via border nodes (nodes lying on the borders between two clusters).
- Asymptotically almost surely (a.a.s) multi-hop network connectivity conditions apply, as assumed in (55).

We consider N nodes deployed in a field with the above assumptions. Each node n_i, where $1 \leq i \leq N$, assigns itself to a cluster c_j, where $1 \leq c_j \leq N_c$ and N_c is the total number of clusters. Note that in all clustering schemes $N \geq N_c$, which is no exception for DVC, however, DVC has an advantage of lower overhead, since it requires no extra communication operations for cluster formation (inherently done by OD-MAC).

4.4.3 Operation

Consider a data gathering scenario where a set of N nodes, within each other's communication range, construct a neighborhood and follow the same wakeup schedule (neighbor discovery follows SMAC (25)). Each node $i \in N$, sets a backoff (BO) timer according to its locally computed LLR, such that $BO \propto 1/|LLR|$ (67). Each node $i \in N$ attempts to send its sensed data D_i to a sink node (base station). Only one node wins the contention and starts transmitting its data to the sink, all the other nodes listen to (overhear) the ongoing transmission then decide whether they need to send their own data. A similarity/tolerance condition, based on which nodes decide whether to send their data packets, is set by the application to reflect the context of interest. That is, if a node determines that it has highly informative information, based on the value of the BO timer (reflecting the LLR value), but it does not satisfy the condition in (Equation 4.1), it will decide *not* to transmit. This will repeat $\forall i \in N$ and is explained in Section 4.4.3.5.

The node winning the contention acts as a CH for all its neighbors that have similar data values, as they will suppress their own data after hearing the contention winner's transmission. All other nodes, not represented by the CH, attempt to transmit their values after the first communication round ends, however, with a higher priority. To avoid deadlocks, if none of the N nodes in the network transmits in communication round C, the first node to acquire the channel in communication round $C + 1$ will unconditionally transmit its locally sensed value.

To illustrate the unique information processing capabilities of the proposed data gathering paradigm, we ignore the LLR operation in this discussion as it has been studied in the literature (67) (68). A detailed example of DVC-DG operation is discussed in Section 4.4.3.5.

To ensure nodes that were not represented by the CH get the priority in accessing the medium, DVC-DG utilizes the Binary Exponential Backoff (BEB) scheme (70) in a reverse manner. The CW is initially set to CW_{max} and divided by 2 upon each retransmission attempt. This will cause nodes that are attempting to transmit for the j^{th} time to have $b \in [0, CW-1]/2^j$, as shown in Figure 16. This will lead to a smaller CW as the number of retransmissions increases, hence, nodes of higher retransmission attempts will have a greater chance of acquiring the channel. The overall operation of DVC-DG is illustrated in Figure 16.

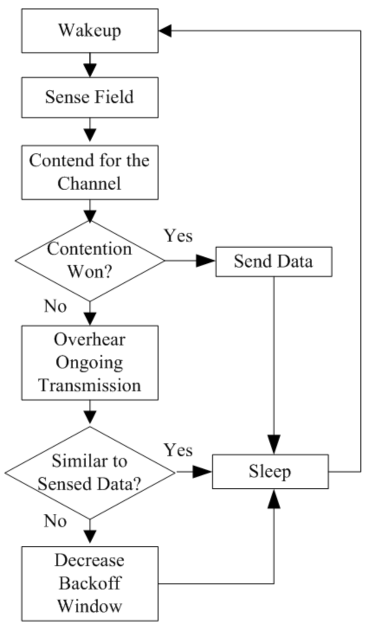


Figure 16. DVC-DG operation

4.4.3.1 Initialization and Connectivity

Each node executes a neighbor discovery algorithm similar to that developed in (25). Initially, each node selects an arbitrary wakeup/sleep schedule to follow. The first node to acquire the channel broadcasts its schedule, and all nodes receiving the broadcast follow it and discard their initially selected schedules. This will lead nodes, within the same neighborhood, to following the same schedule (i.e., initial synchronization) (25) (71). As in all contention-based synchronous MAC schemes, in order to maintain the initial synchronization, periodic synchronization among neighbors is required to overcome clock drift. However, compared to TDMA-like schemes, they require much looser synchronization (8) (25).

A set of nodes within each others' communication range construct a neighborhood. Ideally, each neighborhood should follow the same wakeup schedule, but connectivity cannot be guaranteed. In order to assure connectivity, nodes falling on the peripheries of the communication range of a CH (border nodes) need to adopt more than one schedule as adopted in many neighbor discovery algorithms (25). An alternative approach that has been utilized in the literature is to have border nodes adopt only one schedule and only use the other schedule when multi-hop (inter-cluster) communication is needed (54). However, due to its simplicity and limited control overhead, we adopt the former approach.

4.4.3.2 Utilizing BEB

Binary Exponential Backoff (BEB) has proven to be effective in avoiding collisions in distributed MAC schemes. It reacts to collisions by adjusting the Contention Window *(CW)* size dynamically (70). Before each data transmission attempt, a node senses the channel to determine whether its idle. If sensed idle for a certain duration (DIFS), the node proceeds with its transmission. If the channel is sensed busy, the node defers its transmission until the end of the ongoing transmission and this is when BEB takes place. A backoff timer is initialized to a uniformly chosen value b, $b \in [0, CW - 1]$ and keeps running until it reaches 0 or until the channel is sensed busy, then resumed when the channel is sensed idle again. In the first transmission attempt, each node sets its backoff timer value to CW_{min} and keeps doubling that value every retransmission attempt until a predefined limit of CW_{max} is reached. A packet is dropped after reaching the retransmission limit, R_A, preset by the MAC scheme (70).

DVC utilizes BEB in a reverse manner. The CW is initially set to CW_{max} and divided by 2 upon each retransmission attempt. This will cause nodes that are attempting to transmit for the i^{th} time to have $b \in [0, CW - 1]/2^i$. This will lead to a smaller CW as the number of retransmissions increases, hence, nodes of higher retransmission attempts will have a greater chance of acquiring the channel.

4.4.3.3 Effect of Field Correlation and Similarity

Assume that data sensed at locations X_1 and X_2 has a Mean Square Error (MSE) of $D(X_1, X_2) = 1 - 1e^{-|\alpha|(X_1 - X_2)^2}$ where $|\alpha|$ is the correlation coefficient. Therefore, DVC-DG performs best when the field is highly correlated (lower values of α) and vice versa as illustrated in Figure 17.

DVC-DG achieves similar results of compression techniques used by other clustering schemes (e.g., (54)), without undergoing any actual compression or message exchange. That is, if the field is highly correlated, one CH implicitly represents many nodes in the field which suppress

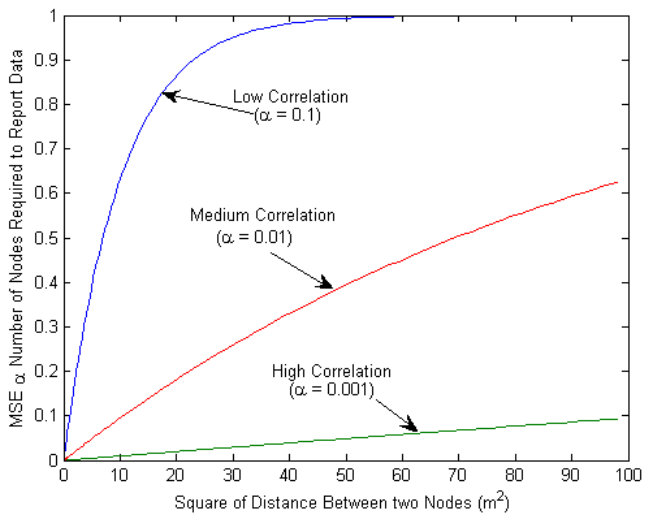


Figure 17. Representing Field Correlation: MSE is proportional to the number of nodes required to report their sensed values. Highly spatially-correlated fields require fewer nodes to report their values and vice versa.

their packets. In DVC-DG, cluster members' main task is to assess the similarity of their data compared to that of the CH rather than to evaluate field correlation (56) or unconditionally send their data to the CH. The worst case scenario for DVC-DG will fall back to conventional data gathering techniques that employ clustering in their operation (e.g., (54), (55), (56)). However, DVC-DG incurs an extra overhearing cost (embedded in OD-MAC operation discussed in Section 4.5.1).

In WSN applications deployed to monitor random fields, similarity in sensed values does not necessarily reflect any spatial correlation (e.g., mine detection in battle fields). To represent similarity in a field, we define a similarity factor F to represent similarity between sensed field values: $F = \frac{1}{N_{Set}}$, $1 \leq N_{Set} \leq N$, where N_{Set} is the number of sets representing the field which has a total of N nodes and S_{V_x} is the set of nodes in a neighborhood with a sensed field value V_x, $min(f) \leq V_x \leq max(f)$, where $min(f)$ and $max(f)$ are the minimum and maximum values of the sensed field, respectively. Note that $\forall V_x \neq V_y$, $S_{V_x} \cap S_{V_y} = \emptyset$. It is clear that the number of sets required to represent a field is tied to the field similarity and DVC-DG can be applied to random fields that assume no spatial or temporal correlation yet have similar uncorrelated values, in such case, the terms correlation and similarity cannot be used interchangeably.

TABLE VI

MESSAGE AND COMMUNICATION COMPLEXITIES

Scheme	Best (*F=1*)	Worst (*F=1/N*)
Generic-DG(e.g., (54; ?))/Random Network	*O(N)*	*O(N)*
DVC-DG/Random Network	*O(1)*	*O(N)*
Generic-DG(e.g., (54; ?))/Binary Tree	*O(N)*	*O(N)*
DVC-DG/Binary Tree	*O(d)*	*O(N)*

Table VI shows the expected message complexities of DVC-DG and context-unaware techniques (e.g., (54), (55)) on two sample networks, a data gathering tree and a random network, each consisting of a total of N nodes. The binary tree is only used for illustration purposes and DVC-DG is independent of network topology. We consider a binary tree of depth d, and nodes of the same depth are assumed to be within each others' communication ranges (i.e., $\in$ the same neighborhood). On the random network, DVC-DG's communication complexity varies from $O(1)$ in the best case to $O(N)$ in the worst case, compared to a constant $O(N)$ in conventional techniques. Also on the binary tree, the lower and upper bounds on communication complexity are $O(d)$ and $O(N)$, respectively. DVC-DG outperforms Generic-DG techniques (i.e., do not target the specific class of data gathering applications we address) that will always require a communication complexity of $O(N)$.

4.4.3.4 Discussion

We highlight and discuss key features in DVC operation via the following observations:

Remark 1: DVC-DG is distributed and requires no topology information. Each node contends for the channel and the winner is the cluster head representing all the nodes with similar values (which choose to be cluster members based on the overheard transmission). Nodes with different values (than that announced by the cluster head) contend for the channel with a higher access probability (smaller CW) and hence get a higher chance of acquiring the channel.

Remark 2: Cluster head election is embedded in the contention-based MAC protocol, therefore, it does not require extra message exchanges and computations. The member nodes of each cluster are responsible for deciding whether they belong to the cluster.

Remark 3: DVC-DG ties communication complexity to field correlation/similarity as illustrated in Figure 17 and Table VI. The higher the field correlation/similarity, the higher the data redundancy and the lower the communication complexity, and vice versa.

Lemma 1: In DVC-DG, a neighborhood with N total nodes and a single sink requires a maximum of $O(1)$ message exchange per node to communicate *all* data to the sink. This is the lower bound of many clustering schemes to only complete the clustering process (i.e., without data aggregation and reporting to the sink) (53) (55). This is due to DVC-DG's implicit cluster formation (via overhearing), which eliminates unconditional message exchange.

Proof. Let N be the number of nodes in a neighborhood (recall that a neighborhood is a set of nodes within each other's communication range). Since the maximum communication requirement will occur when each node in the neighborhood senses a different value (the sensed field is completely uncorrelated "α=1"), therefore a total of N messages is required to be communicated to the sink i.e., one message per node.

Lemma 2: DVC-DG requires a single message exchange/neighborhood in a fully correlated field.

Proof. In fully/highly correlated fields, where $\alpha \to \infty$ and $D(X_1, X_2) \to 0$, all nodes will sense similar values. Therefore, the first node acquiring the channel (cluster head) completely represents the entire neighborhood, and consequently, all members suppress their pending transmissions.

Lemma 3: DVC-DG's execution includes implicit cluster formation, message aggregation and delivery to sink, all combined in $O(1)$ iterations.

Proof. Upon every attempt to acquire the channel, a node reduces b by half until it reaches 0. The initial value of b is set between $[0, CW_{max} - 1]$, hence the max number of iterations per communication round (per frame) is $\log_2(CW_{max} - 1)$.

Note that other clustering-based data gathering techniques terminate in constant time, however, they do not link communication operations to field correlations/similarities, thus data redundancy is not reflected in the number of messages exchanged.

4.4.3.5 Example

The operation of DVC-DG can be further illustrated via the following example. Consider a random network with a single sink node and a total of N nodes randomly distributed over an area of radius R equal to half the communication range of a node's radio, as shown in Figure 18(a). Each node color in Figure 18 refers to a certain sensed value, and data similarity is based on satisfying a certain accuracy/tolerance condition set by the application.

Initially, nodes are not aware of such similarities in their sensed data and all of them contend for the channel to report their data to the sink. Only one node wins the contention (the CH) and transmits its data packet to the sink, while other nodes overhear the transmission, as shown in Figure 18(b). Since all blue nodes have already been represented by the CH, there is no need for them to transmit their values. Nodes that have not been represented in the current communication round increase their chance of acquiring the channel as discussed in Section 4.4.3 and shown in Figure 18(c) (darker nodes).

4.5 Analysis of OD-MAC

4.5.1 Operation

Most MAC protocols for WSNs either try to avoid overhearing or discard overheard packets without further processing (25) (71). The actual source of energy waste is the *unconditional* negligence of overheard packets. As shown in section 4.4.3.4 above and proposed in recent stud-

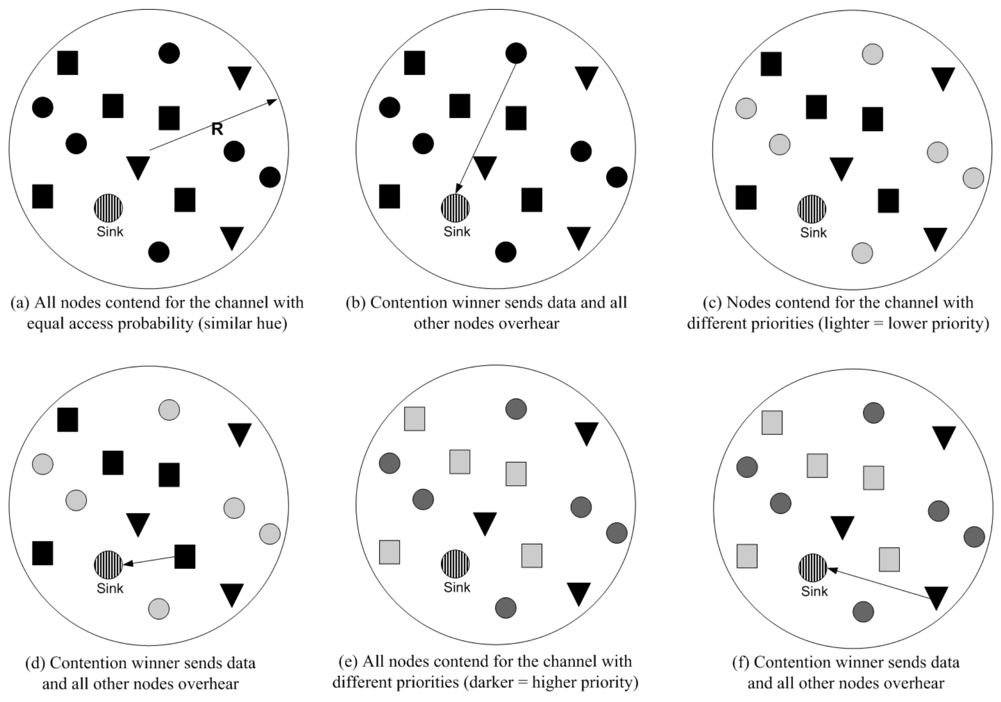

R
Sink
(a) All nodes contend for the channel with equal access probability (similar hue)
Sink
(b) Contention winner sends data and all other nodes overhear
Sink
(c) Nodes contend for the channel with different priorities (lighter = lower priority)
Sink
(d) Contention winner sends data and all other nodes overhear
Sink
(e) All nodes contend for the channel with different priorities (darker = higher priority)
Sink
(f) Contention winner sends data and all other nodes overhear

Figure 18. DVC-DG operation example: different node shapes reflect different sensed values. Darker hue reflects higher channel access priority, and vice versa

ies (58) (59), overhearing can greatly reduce communication redundancy. Processing overheard packets informs the node of what information is being communicated over the channel from other nodes in its communication neighborhood. This can be particularly beneficial to data gathering and monitoring applications as illustrated in section 4.4.3.4.

We modify S-MAC's (25) operation to induce additional overhearing necessary for DVC-DG's functionality, and we call it OD-MAC (Overhearing Dependent MAC). As in all synchronous contention-based MAC schemes, neighboring nodes in OD-MAC coordinate their wakeup/sleep schedules (8). However, nodes that loose the contention do not go to sleep

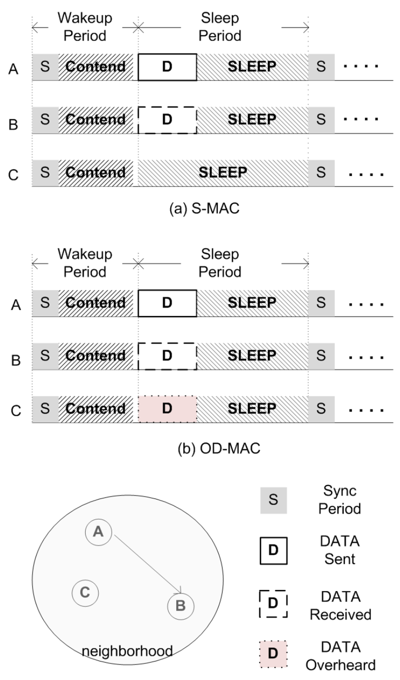


Figure 19. Operation of OD-MAC and S-MAC

immediately (as in S-MAC), but rather stay awake to overhear the ongoing transmission of the contention winner, as shown in Figure 19. This introduces additional overhearing-generated energy consumption that may or may not be beneficial (i.e., depending on the similarity between the overheard packet and the pending packet). This is clearly a trade-off between extra energy consumed in overhearing and that saved by avoiding unnecessary transmissions, which is dependent on the similarity in the sensed data. Therefore, the energy consumed in overhearing needs to be studied along with the similarity of sensed data, as discussed below in Section 4.5.2.

4.5.2 Overhearing Cost

Since we are assuming a customized MAC protocol (OD-MAC) at each node, in this section we present the analysis of the cost of additional overhearing compared to other contention based MAC protocols, namely S-MAC (25) and LWT-MAC (59). We also study the energy breakdown of OD-MAC in comparison to the other two schemes in order to quantify the cost of additional overhearing incurred by OD-MAC. We follow the analysis and channel assumptions in (70), which are also adopted in (59). For comparison fairness, we compare OD-MAC to the scheduled mode of LWT-MAC (59). We formally present different sources of energy consumption and their dependence on network and application parameters, such as collisions and queue utilization.

In duty-cycled MAC schemes for WSNs, there are two main states for a node: active and sleep. During its active state, a node can transmit, receive or listen to the channel, and during its sleep state the node turns off its radio. Each node is assumed to have a queue of finite length Q, and each packet in the queue has an average length L_{DATA} bits. We assume that data packets are generated following a Poisson process with a rate equal to λ packets/second (i.e., inter-packet times are independent and have an exponential distribution with a mean $= 1/\lambda$). However, more complex traffic models can also benefit from our technique but with different distributions of trade-offs between energy consumed in overhearing and that saved from collision avoidance and conditional message transmissions. Each packet is assumed to spend an average of T_{delay} before leaving the queue, which is the sum of queuing delay and service time computed by:

$$T_{delay} = \tau_s N_s + (A-1)(\tau_s N_s + T_C) \tag{4.4}$$

where τ_s is the average slot duration, N_s is the average number of slots skipped before acquiring the channel on each transmission attempt (Back off window), A is the average number of transmission attempts needed per packet, and T_C is a collision duration. A can be represented as a function of the collision probability P_C and the maximum number of retransmission attempts R_A such that $A = \frac{(1-P_C^{(R_A+1)})}{1-P_c}$. The collision probability is related to the number of nodes in the network, N, where $P_C = 1-(1-P_r)^{N-1}$ and P_r is the probability of a node, having a packet ready to be sent, to transmit in a random slot. P_r can be related to the queue utilization factor ρ by $P_r = \rho/(N_s+1)$, where $\rho = \lambda/\mu$ and μ is the mean service time. Like in S-MAC, T_C is deduced from IEEE 802.11 as well as the values of the guard periods, SIFS, DIFS and EIFS (25). $T_C = DIFS + SIFS + L_{RTS}/r$ and $T_S = \frac{L_{RTS}+L_{CTS}+L_{DATA}+L_{ACK}}{r} + DIFS + 3SIFS$ is the time needed to successfully transmit one data packet, and r is the transmission rate. Note that for a uniformly distributed back off window over the maximum contention window will lead to $N_s = \frac{CW_{max}}{2}$. The throughput of the queue can be computed as $\gamma = \lambda(1-P_B)$ and $P_B = \frac{(1-\rho)\rho^Q}{1-\rho^{Q+1}}$ is the blocking probability (i.e., probability that the buffer is full). A list of notations frequently used in describing DVC-DG functionality are listed in Table VII.

We assume possible channel states with respect to the sending node to be: (a) empty (neighbor nodes are idle listening or sleeping), (b) sending/receiving, and (c) collision. Each state has a corresponding probability of (a) $P_e = (1-P_r)^{N-1}$, (b) $P_{s/r} = P_r(N-1)(1-P_r)^{N-2}$, and (c) $P_c = 1 - P_{s/r} - P_e$, respectively. The total energy consumption of a node is due to transmitting, receiving and overhearing, and each one of these energy components has a certain successful and collision component in it. That is, a transmission, reception, or overhearing can either be successful or not, based on collisions. This leads to:

$$E_{total} = E^s_{tx} + E^c_{tx} + E^s_{rx} + E^c_{rx} + E^s_{oh} + E^c_{oh} \tag{4.5}$$

where E^s_{tx}, E^s_{rx}, and E^s_{oh} are the energies consumed in successful transmission, reception and overhearing, respectively. E^c_{tx}, E^c_{rx} and E^c_{oh} are the energies consumed in collided (unsuccessful) transmission, reception and overhearing, respectively. The value of each one of these energy components will vary according to the MAC protocol behavior. Each of the above energy components suffers an amount of idle listening as well (e.g., during DIFS and SIFS). We refer to the energy consumed in a node's radio states, transmission, reception and idle as E^{TX}_{radio}, E^{RX}_{radio}, and E^{IDLE}_{radio}, respectively. Radio sleep state is assumed to consume no energy (59) (70), and therefore:

$$E_{tx}^{s} = E_{radio}^{TX}\frac{L_{RTS} + L_{DATA}}{r} + E_{radio}^{RX}\frac{L_{CTS} + L_{ACK}}{r} + E_{radio}^{IDLE}(DIFS + 3SIFS + N_s P_e \epsilon) \quad (4.6)$$

$$E_{tx}^{c} = E_{radio}^{TX}\frac{L_{RTS}}{r} + E_{radio}^{IDLE}(DIFS + 2SIFS + N_s P_e \epsilon + \frac{L_{CTS}}{r}) \quad (4.7)$$

where ϵ is the duration of an empty slot. The energy consumed in successful and unsuccessful receptions can be represented by Equation 4.8 and Equation 4.9, respectively.

$$E_{rx}^{s} = E_{radio}^{RX}\frac{L_{RTS} + L_{DATA}}{r} + E_{radio}^{TX}\frac{L_{CTS} + L_{ACK}}{r} + E_{radio}^{IDLE}(3SIFS) \quad (4.8)$$

$$E_{rx}^{c} = E_{tx}^{c} - (E_{radio}^{TX} - E_{radio}^{RX})\frac{L_{RTS}}{r} \quad (4.9)$$

Since overhearing occurring in conventional MAC protocols is mainly that of control packets, this does not serve our purpose. We are interested in overhearing data packets regardless of

TABLE VII

NOTATIONS USED IN DESCRIBING DVC-DG FUNCTIONALITY

Symbol / Abbreviation	Description
CH	Cluster Head.
DVC	Dynamic Virtual Clusters.
DG	Data Gathering.
BO	Back-Off.
LLR	Log-Likelihood Ratio.
BEB	Binary Exponential Back-off.
CW	Contention Window.
MSE	Mean Square Error.
α	Correlation Coefficient.
L_{DATA}	Length of Data packet.
Q	Length of the queue.
λ	Rate of Poisson process.
P_e	Probability of channel being empty.
$P_{s/r}$	Probability of channel being occupied by a successful transmission/reception.
P_c	Probability of channel being occupied by a collision.
P_C	Collision Probability.
P_B	Blocking Probability.
P_r	Probability of having a packet ready to be sent.
T_C	Collision Duration.
R_A	Maximum number of Retransmission Attempts.
ρ	Queue utilization factor.
τ_S	Average slot duration.
N_S	Average number slots skipped before acquiring the channel.
RTS	Request To Send.
$SIFS$	Short Inter-frame Space.
$DIFS$	Distributed Inter-frame Space.
$EIFS$	Extended Inter-frame Space.
$E^S_{\cdots}$	Energy consumed in a successful operation.
$E^C_{\cdots}$	Energy consumed in an unsuccessful operation (collision).

their source. This is the overhead OD-MAC encounters, as data packets are much larger than control packets. Overhearing in OD-MAC can be expressed as:

$$E^{s}_{oh} = E^{RX}_{radio} \frac{L_{DATA} + L_{RTS}}{r} + E^{IDLE}_{radio} CW \tag{4.10}$$

$$E^{c}_{oh} = E^{RX}_{radio} \frac{L_{DATA} + L_{RTS}}{r} + E^{IDLE}_{radio} (2SIFS + \frac{L_{CTS}}{r}) \tag{4.11}$$

4.6 Performance Evaluation

In this section, we evaluate DVC-DG based on the analysis in Section 4.4.3.3, Section 4.5.2, and NS-2 simulations. The main goal is to examine DVC-DG's performance compared to other Generic Data Gathering (Generic-DG) techniques, which do not target any specific type of data gathering applications (54) (55) (59).

We compare the message complexity of DVC-DG to that of generic techniques (54) (55), under various fields and applications, and compare analytical results to those from simulations. Since DVC-DG's performance is tied to the similarity in sensed data *(F)*, we study the trade-off between the energy saved from message/collision reduction and that consumed in extra overhearing. We compare the total energy consumption of DVC-DG (employing OD-MAC) to that of Generic-DG techniques utilizing S-MAC (25) and LWT-MAC (59), under different operating conditions. In addition, we study the energy breakdown to show how much each component contributes to the overall energy consumption, and show the effect of queue utilization (proportional to data rate) on overhearing and collisions in all three techniques.

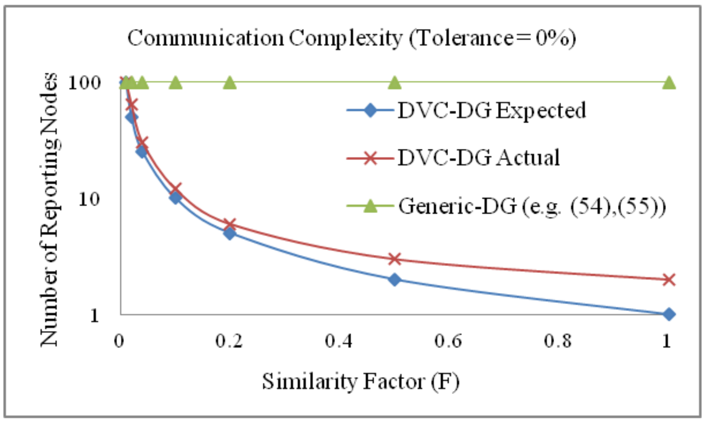
TABLE VIII

SIMULATION PARAMETERS (DVC-DG)

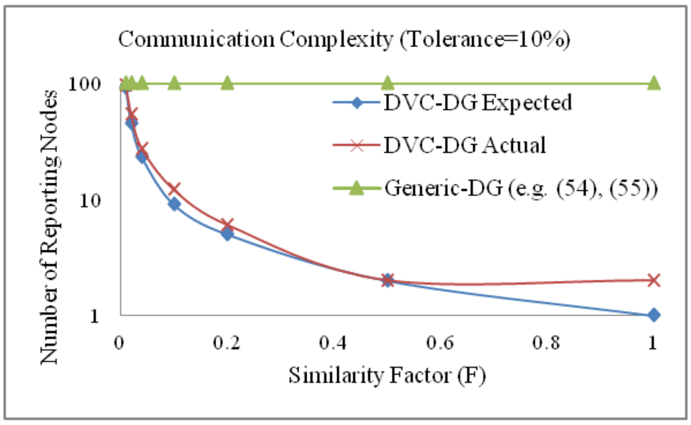
Parameter	Value	Parameter	Value
Bandwidth	20 kbps	*Comm. Range*	250 m
RxPower	22.2 mW	*Interference Range*	550 m
TxPower	31.2 mW	*DIFS*	10 ms
IdlePower	22.2 mW	*SIFS*	5 ms
SleepPower	12 µW	*Contention Window*	64 ms
DataPckt	30 B	*MAC scheme cycle*	4544 ms
ACK	8 B	*Duty Cycle)*	33%

We simulate a 100-node randomly deployed network sensing a field, covering a total area of $10,000\,\text{m}^2$ ($< \pi.R^2$). The network has a randomly selected sink to which 50% of the nodes are required to report their values. However, this percentage is not fixed for DVC-DG as its operation is dependent on data similarity (see Table VI). For comparison fairness, we compare OD-MAC (employed by DVC-DG) and (25) to the scheduled mode of (59). In order not to favor DVC-DG by choosing fully/highly correlated fields, we simulate a wide range of fields with various similarities/correlations. In particular, we vary F between 0 (uncorrelated field) and 1 (fully correlated field) with an 0.05 step size and report the average of all fields. Each simulation is an average of 30 runs, each lasting for 6000 seconds. Key network simulation parameters are summarized in Table VIII.

The above simulation conditions remain unchanged for all proceeding results, except for Figure 23, where we explore the *break-even point*. That is, the point at which the energy



(a)



(b)

Figure 20. Comparing communication complexity of DVC-DG to Generic-DG techniques

savings from communication and collision reduction are offset by the energy consumed in the extra overhearing operations occurring in DVC-DG.

Figure 20 shows the difference between the expected number of messages to be communicated over the channel (according to Table VI) and the actual simulation results at different application tolerance levels. The different tolerance levels can result from either two different applications or the same application under different constraints. When the tolerance (set by the application) is 0% as shown in Figure 20(a), the average error (difference between the expected number of reporting nodes and the actual number of reporting nodes) is 12.7%. This is expected due to wireless channel variations and limitations of the underlying communication protocol (e.g., collisions). Since techniques that rely solely on spatial and/or temporal field correlations in their operation (e.g., (59) (56)) do not realize field similarity occurring *only* in the data space, their message complexity is not affected by uncorrelated field similarity and remains at its maximum. Figure 20(b) reflects the result of increasing the tolerance to 10%, which decreases the average number of reporting nodes by approximately 10% as well, and the average error remains at 12%. These low error values reflect the high quality of information gathered at the sink and verify the accuracy of the analysis in Section 4.4.3.3.

DVC-DG consumes extra energy in overhearing other transmissions, and it can comprise up to 54% of the total energy consumed as shown in Figure 21. Although generic data gathering techniques employing (S-MAC and LWT-MAC) do not benefit from overhearing in their operation as DVC-DG does, they still waste up to 30% of their total energy on unavoidable overhearing. From Figure 21, it is clear that the variation in overhearing energy with data rate is insignificant when employing LWT-MAC and S-MAC in data gathering. This is due to the variation in overheard control overhead, which is considerably lower than the overhead of overhearing data. DVC-DG (employing OD-MAC) consumes more energy because nodes stay awake after losing the contention to overhear the CH transmission. The extra energy consumed by DVC-DG is solely due to the added overhearing and can be offset depending on the sensed

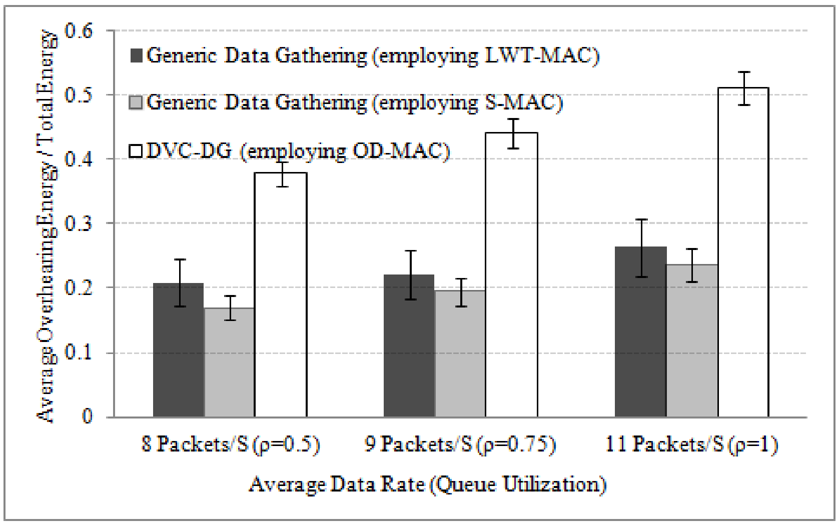


Figure 21. Overhearing energy to overall energy (normalized to the average total energy consumption)

field correlation/similarity. Error bars in Figure 21 and Figure 22 indicate the maximum and minimum values of all runs.

Figure 22 shows that, on the average, DVC-DG (employing OD-MAC), S-MAC and LWT-MAC consume 27%, 53% and 56% of their total energy on unavoidable collisions, respectively. The reason DVC-DG suffers nearly half the collisions lies in its ability to reduce traffic on the network by overhearing other nodes transmissions. Note that DVC-DG's collision reduction comes at a cost of extra overhearing, and hence the total energy savings are less than the energy saved by collision reduction. This trade-off is further discussed below.

Figure 23 illustrates the energy breakdown of DVC-DG and Generic Data Gathering (Generic-DG) via LWT-MAC (59) and S-MAC (25), at different data rates (different queue utilizations) and for different field similarity factors (F). Note that many physical phenomena (e.g., temper-

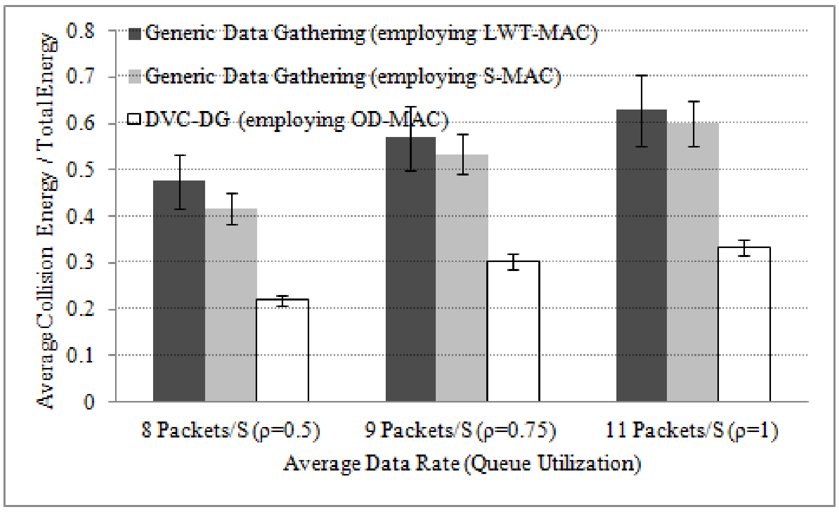
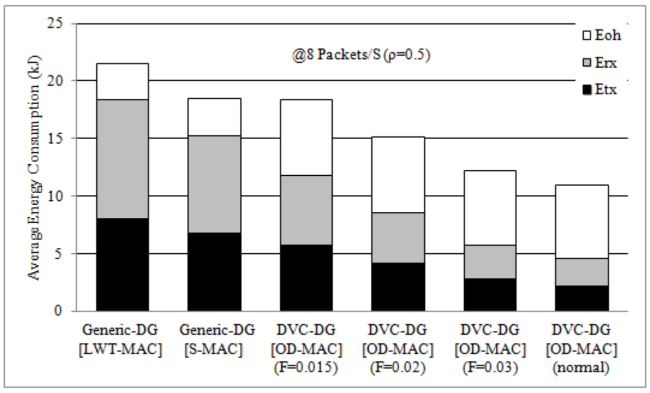


Figure 22. Effect of collisions (normalized to the average total energy consumption)

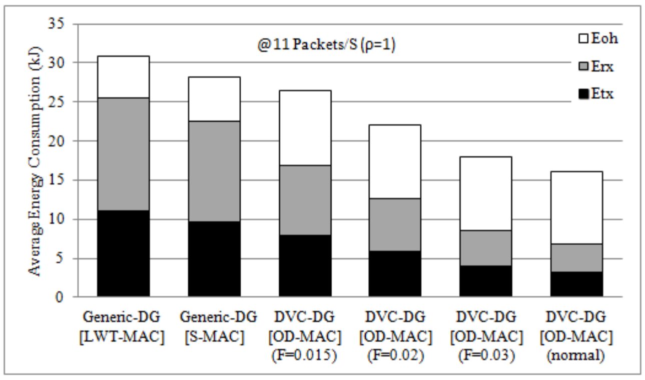
ature) can be represented by a standard normal distribution, which has data similarity of a field with $F \approx 0.04$. Hence, fields with such characteristics are referred to as *normal* in Figure 23.

From Figure 23 (a) and (b), it is evident that the break-even point, where DVC-DG's energy savings are offset by the extra overhearing energy consumption, occurs at a rather low value of $F \approx 0.015$. That is, if 67% (or less) of the total nodes must transmit their sensed data, DVC-DG works effectively and achieves overall energy savings. Note that Generic-DG are not sensitive to changes in F

As shown in Figure 23 (a), for the normal field case ($F \approx 0.04$), DVC-DG consumes an average of 47% and 42% less energy, compared to (59) and (25), respectively. Comparing Figure 23 (a) and (b), it is clear that increasing queue utilization affects DVC-DG's energy savings. This is in part due to more collisions resulting from higher data rates, which affect



(a)



(b)

Figure 23. Breakdown of the average energy consumed by DVC-DG compared to other techniques. E_{tx}, E_{rx}, and E_{oh} are the energies consumed in transmitting, receiving, and overhearing, respectively. Normal refers to normal standard fields ($F \approx 0.04$).

successful overhearing of packets. In addition, the packet drop rate increases as nodes receive packets at a higher rate than they can process, and will eventually drop packets (queue overflow).

4.7 Summary

In this chapter we presented a DVC-DG, CL distributed data gathering technique for reducing communication overheads in data gathering and monitoring applications in WSNs. It utilizes overhearing (at MAC layer) to eliminate communication redundancy during cluster formation and operation (application layer). Our approach takes advantage of medium access contentions to implicitly elect representative nodes (CHs). After acquiring the channel, the CH transmits its sensed value to the sink while all its neighbors overhear the transmission. If the CH's overheard data is similar to what a node currently senses, this node suppresses its packet since it was already represented at the CH, and becomes a cluster-member. Otherwise, it contends for channel access in the following cycle. Processing overheard packets eliminates the root cause of data redundancy; hence, improving communication efficiency. It also binds the number of message exchanges within a cluster to the data similarity present in it. This creates an on-demand communication state, where nodes only transmit when necessary.

Based on our analysis and simulations, we showed that the overhearing-introduced penalty can be offset by moderate similarities and correlations in the sensed field. Although DVC-DG mainly targets applications that do not require entire field recovery, it can perform well in such applications when the field is relatively static, and an aggregate value is sufficient. DVC-DG can reduce the number of messages communicated in a neighborhood consisting of N nodes, by up to $N-1$.

CHAPTER 5

A CROSS-LAYER DESIGN PARADIGM AND SIMULATION INTERFACE

5.1 Introduction

As mentioned in section 3.1 above, a wide variety of CL protocols exists in the literature, varying *from* utilizing CL information (information from a different layer) in optimizing the operation of another layer (10) *to* completely merging the functionality of several layers (72) (73). Along with CL protocols, CL paradigms, aiming at standardizing the CL design process, also emerged (46) (74) (75) (76). Since most CL protocols do not follow a specific CL paradigm (10) (40), and many CL paradigms are tailored towards specific protocol optimizations (77) (51), in our brief review of those efforts, we refer to both as CL approaches.

CL approaches can be classified, based on their architecture and behavior, as being evolutionary or revolutionary (78). Evolutionary approaches consider compatibility with the standard OSI model (i.e., incorporating CL capability into the OSI architecture), while revolutionary approaches only consider performance optimization and completely disregard the standard (i.e., several layers are melted into one). Examples of evolutionary CL approaches in Mobile Ad hoc Networks (MANETs) were presented in (46) (74) (75) (76) (78), and other studies adopting the revolutionary approach WSNs are discussed in (72) and (79). A key difference between

MANETs and WSNs, that must be considered in any CL approach for WSNs, is that the latter suffers much tighter energy constraints, due to infeasibility of battery recharging.

In this chapter, we propose a Cross-Layer Application-aware Paradigm (CLAP), to facilitate CL protocol design in WSNs and overcome implementation complexities encountered in most CL approaches. CLAP incorporates an Information-Layer (I-Layer) as means of CL interaction. The I-Layer is accessible to all layers of the network stack through a *publish*-and-*subscribe* fashion. The I-Layer's architecture gives the application layer the capability of directly accessing lower layer(s) information and modifying their behavior(s), without following the conventional OSI hierarchy. This introduces a new level of application layer awareness and control over underlying protocols, which is a key distinction between CLAP and other CL approaches.

We implement an extension to the SIDnet-SWANS simulator (80) to incorporate CLAP. By designing the I-Layer as a hash table and providing an API for other layers to store and access information in it, we are able to realize the proposed paradigm in this simulator. Based on the simulator, we also demonstrate a sample scenario adopting the new paradigm. CLAP increases the level of adaption and awareness, while simplifying the design, implementation and operation of CL protocols. It allows changes made at the very top of the network stack to directly impact its very bottom, which gives CL protocol developers unprecedented design freedom and implementation simplicity.

The remainder of this chapter is organized as follows: section 5.2 is a brief review of CL protocols and paradigms. Section 5.3 discusses the details of CLAP, its architecture and implementation which extends the SIDnet-SWANS simulator. An example CL protocol, utilizing CLAP, is presented in section 5.4, followed by a summary in section 5.5.

5.2 Background

It is suggested by many studies that the revolutionary CL approach is more suitable for emerging technologies, such as WSNs (78) (45). This is true when such systems require minimal compatibility and operate in a standalone fashion. However, many WSN applications are integrated and interfaced with larger, conventionally designed networks (e.g., (24)). For example, consider a modern health care system, where wireless sensors attached to a patient's body, forming a WSN. The sensors monitor different vital signs (e.g., blood pressure, heart rate, temperature etc.) and periodically report their data to the patient's doctor and send alerts during an emergency. To achieve this, the WSN must be interfaced to the local cellular network, which is conventionally designed and operated. This requires the WSN to communicate with the patient's cellular device, which will forward the sensory data to the doctor via text messages/e-mail. This emphasizes how a WSN can indeed be part of a larger network that is conventionally designed (OSI-like architecture) and operated. Therefore, sacrificing interoperability and compatibility, as suggested in revolutionary CL approaches, must be reconsidered in such scenarios. We use the terms conventionally and OSI-Like interchangeably.

Realizing limitations due to absence of modularity in revolutionary CL approaches, many research efforts adopted the evolutionary approach in proposing CL paradigms. In (75), CL information reflects a layer's state and can only be communicated between adjacent layers. Attempting to regulate CL interactions, information from non-adjacent layers is accessed via a mapping of that information to the adjacent layer. In (76), CL information is stored and organized via a network status entity that preserves layer separation. Finally, in (74), a similar architecture to that in (76) is proposed, however global network information is piggybacked over each data packet and factored in the optimization decision. All the above CL paradigms and others, such as (81) and (82), mainly target MANET environments with less critical energy constraints, compared to WSNs, and do not give in-depth details about an actual protocol operation. Moreover, they treat the application layer as a client that demands service without being involved in operation of lower layers. A comparison between several CL paradigms is presented in (74).

In addition to the above studies proposing new CL paradigms, other efforts developing CL protocols, not adhering to any CL paradigm, also exist. These efforts vary *from* utilizing a single parameter from one layer in the functionality of another *to* merging the functionality of two layers (45) (8). For example, as discussed above in section 2.2, next-hop information from the routing layer is utilized at the MAC layer to schedule multi-hop traffic flows (10). The shared information was extended in (40), where all packets in the routing buffer were considered in the scheduling process and led to significant performance gains. On the other hand, in (73), a joint scheduling and routing scheme was proposed (i.e., completely melting routing and MAC). The main differences between CL protocols are (i) the type and amount of CL information communicated/shared and involved in the design, and (ii) which layers interact and how.

Although CL information utilized is different and for different purposes, the evolutionary CL protocols discussed above, and many others summarized in (8) (45), share many of the following design and implementation aspects and limitations:

- expose CL information, that is hidden or considered irrelevant in conventional designs, which led to significant performance improvements.
- CL interaction is usually tailored to one layer and a few parameters (e.g., packet similarity/order in routing buffer in (40))
- application layer interaction/involvement in lower layer operation is very limited (e.g., overheard packets in (83)).
- were implemented via CL operations on simulators designed with OSI-like architectures at the core of their operation, which influenced the CL design and significantly increased implementation complexity.

The design complexity of CL protocols is partly due to incorporating CL information in the design process; however, a greater complexity lies in the implementation and instability of such protocols. The implementation complexity results from the need to *bypass/hack* simulator hierarchies that originally targeted non-CL simulations. This strongly contradicts the main purpose of discrete-event simulations which aim at reducing complexity (84). This is also a main reason for the limited presence of CL paradigms in the design process of various CL protocols (i.e., limited publicity of CL paradigms beyond the proposing research group).The instability encountered in any CL approach results from, often foreseen, joint optimization of layers. That is, feedback from one layer affects the operation of another, which creates a closed-loop feedback system (85).

5.3 CLAP Design

In this section, we illustrate CLAP's design by comparing it to conventional paradigms and emphasize its' novelty compared to state of the art in CL design. We also cover CLAP's implementation details which extends SIDnet-SWANS (80).

5.3.1 Comparing to Conventional Designs

We present a new CL paradigm to facilitate the design of CL protocols for WSNs in which CL interaction and control is achieved via the I-Layer (Information-Layer). The I-Layer is accessible to all other layers of the network stack and contains all information shared between them. The information stored in the I-Layer can either be status or control information. One of the layers (the application layer in the proposed scenario) is granted control over CL operation and hence dictates how the information is utilized (e.g., trade-off energy consumption for throughput). The controlling layer publishes control information, reflecting the desired behavior, to which other (non-controlling) layers subscribe and adapt their behavior accordingly. Non-controlling layers publish status information, to which the controlling layer subscribes, to retrieve as discussed in subsection 5.3.2.

Figure 24 illustrates the main differences between conventional (OSI-like) CL information sharing/interaction and CLAP. For presentation simplicity, we integrate the transport layer into the routing layer and the physical layer into the MAC layer. In CLAP, assuming application control of CL interaction as in Figure 24(b), I_A captures the application decision to modify the

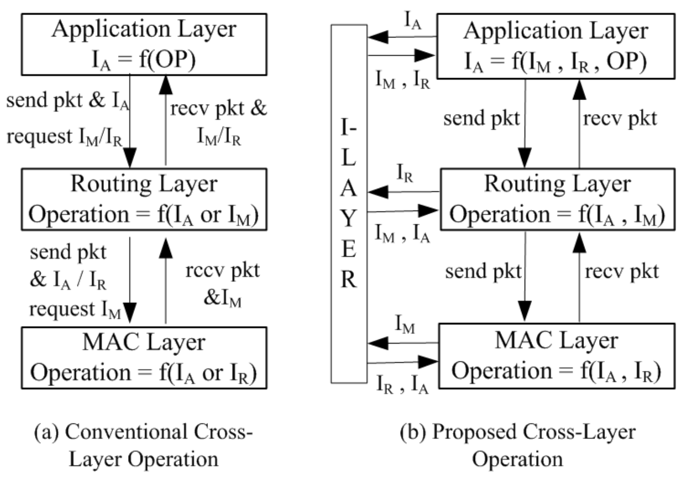


Figure 24. A comparison between CL protocol operation in (a) conventional paradigm and (b) CLAP: I_A, I_R and I_M denote information shared by Application, Routing and MAC layers, respectively. The functionality of each layer is denoted by f( ), and OP represents Optimization Parameters

underlying MAC and/or routing operation. However, in the conventional case, I_A only represents CL info (i.e., no control info). In CLAP, CL information is communicated and handled via the I-layer, which allows non-adjacent layers to interact without *unnecessary* involvement of other layers. For example, the application and MAC layer can interact without routing layer involvement. This would greatly simplify the design and implementation of protocols involving such CL interaction. In (83), the overheard data packets (at the MAC layer) are processed at the application layer which suppresses packets to be sent, if it detects correlation between them and the overheard packet. In addition, accessing other layers' information does not require permission nor hacking the design hierarchy, it is accessible via the I-Layer. Another advantage of CLAP is the ability of involving more than two layers in the interaction and no restriction on what information can be shared. This makes tri-layer interaction possible as discussed in section 5.4.

Although the idea of providing means of CL interaction, involving more than two layers, is not new (76) (86), the application's involvement/control capabilities over the protocol as well as the information sharing technique are unique to CLAP. Moreover, studies like (76) (86) mainly target MANETS, where energy constraints are considerably looser (i.e., no duty cycling), compared to WSNs, and hence, suffer less performance instability (87).

To demonstrate the unique capabilities of CLAP, we consider the case of application control over protocol behavior. Such control is embedded in CL control information (I_A in Figure 24(b)) and its details are decided by the protocol designer (see section 5.4 for an example). This gives the application knowledge about underlying layers' operating conditions and direct control over their operation. For example, in a conventional scenario, the running application is not aware and has no control over the transmission *retry limit* which is solely set by the underlying MAC scheme (retry limit defined and illustrated in section 5.4). In CLAP, the application layer would not only be aware of the retry limit set by the MAC protocol, it would have the ability to change it according to its observations (I_M and I_R) and its needs (reflected in OP), resulting in an application-controlled customizable and reconfigurable MAC operation. More details about application awareness, optimization parameter formulation and interaction with other layers can be found in (51).

Notice that all protocols in the literature, where CL interaction is limited to information sharing (e.g., MAC and routing interaction in (40)), can still be implemented using CLAP. In such scenario, the controlling and controlled layer are the same (i.e., no control info will be published). However, more efficient and seamless access to CL information can be achieved via the I-Layer (i.e., no need for extra control messages to request access to CL info and no complex hacking for simulator hierarchy). Moreover, CLAP can implement non-CL protocols, which guarantees compatibility, however at the cost of the non-utilized CL capability overhead (i.e., I-Layer).

5.3.2 CLAP Implementation

CLAP's novelty lies in the following unique features:

- CL information is encapsulated in the I-Layer via a *publish*-and-*subscribe* manner, which eases CL information sharing by enabling direct interaction between any two layers without having to go through others.
- allowing all layers *(including the application layer)* to publish and subscribe to the I-Layer, which makes complex CL interactions, especially those involving more than two layers, easier to control, manage, and implement.

5.3.2.1 Architecture

Selecting the simulation platform to which we augment our work was key to achieving our goal of simplifying CL implementations. Several open source simulation tools have gained great attention and are being utilized by the research community in evaluating wireless sensor and ad hoc networks (44) (88) (89) (90) (91). Various studies comparing these tools, particularly NS-2 and JiST/SWANS, have showcased JiST/SWANS' superior performance which allows for more detailed simulations in shorter times (92) (93) (94). This speedup is mainly due to JiST/SWANS' very low memory footprint which stems from its ability to use Timeless Objects (i.e., passed by references rather than by copy), which enables large savings in memory usage compared to NS-2 where objects are copied far more aggressively in order to protect the integrity of the simulation. Nevertheless, each particular implementation will have implications on the simulator performance.

SIDnet is a simulation-based environment which enables run-time interactions with the network both at an individual node, as well as a collection of (80). For all these reasons, we chose to implement our work on SIDnet-SWANS. Figure 25 depicts the architecture of the system and our extensions are highlighted. SWANS provides the implementation to represent the network stack of each node in the simulation, while SIDnet provides a user-friendly GUI together with a NodeAPI, giving the user run-time information, to better manage all the nodes (80).

Our implementation utilizes the advantages of SIDnet-SWANS architecture, and supports CL interaction by extending the behavior of the node stack and NodeAPI, in order to integrate the I-Layer into the system. Layers that are involved in CL interactions will publish their CL information into the I-Layer and subscribe to CL information of interest that is published by other layers. The I-Layer itself acts as storage for CL information and demands. This is achieved via the publish and subscribe actions of other layers. In the remaining part of this section, we will discuss those functionalities in detail.

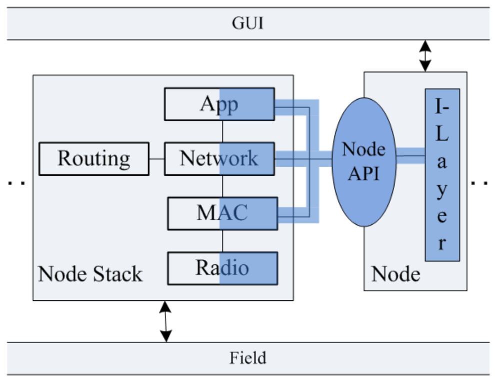


Figure 25. System Architecture: Modifying CL interaction in the node stack by enabling CL information sharing via the new I-Layer through an extended NodeAPI interface. Extensions are integrated into SWANS' node stack and interfaced to SIDnet's GUI and NodeAPI. Modifications/extensions are highlighted

5.3.2.2 I-Layer Hookup

Being a layer that is not part of the traditional network stack, the hookup of the I-Layer can be non-trivial. However, the implementation of SIDnet-SWANS makes this task relatively straight forward. From Figure 25 we can see that, for each node, all the layers involved in CL interactions are associated within the *Node* object. In the implementation of SIDnet-SWANS (80), the class defining each layer includes the Node object as an instance variable. This makes the Node class a reasonable choice for placing the I-Layer, since all the layers will have an easy access to the I-Layer through the Node object. In our implementation, we assign an *I-Layer* object as an instance variable in the Node class. We also extend the methods in the *NodeAPI*, and the interface of the Node class to expose methods interacting with the I-Layer. Hence, through the NodeAPI, all the layers involved in the CL interaction will have a unified access to the I-Layer. The details of how this access occurs is discussed in subsections 5.4.2 and 5.4.3.

5.3.2.3 Publishing CL Information

Before publishing CL information into the I-Layer, each type of information has to be assigned a key which has to be agreed on among all layers taking part in the CL interaction. When some layer l wants to publish CL information with key k, it calls the *publish(key)* method in the NodeAPI as shown in Figure 26. This will trigger the control unit of I-Layer to put a new key-value pair with key k in the internal hash table of the I-Layer. The control unit will also remember that the entry with key k in the hash table can only be updated by layer l. When layer l has the actual data to write into I-Layer, it will call the *update(key, value)* method in NodeAPI. This will trigger the control unit to verify the caller being layer l and update the entry with key k in the hash table.

5.3.2.4 Subscribing to CL Information

CL information sharing not only requires publishing information to the I-Layer, but also subscribing to the I-Layer in order to access other layers' shared information. The subscribing part of the interaction works as follows. Since the layers have agreed upon the keys for all CL information, if layer l wants to subscribe to a particular type of information with key k, it only needs to call the *subscribe(key)* method in NodeAPI (Figure 26). However, before that, it also needs to call the *getKeyList( )* method in NodeAPI to verify that the information has already been published. *getKeyList( )* tells control unit of the I-Layer to return all the keys in the hash table as a list. The caller, layer l, will then be able to see whether k is within this list. If the

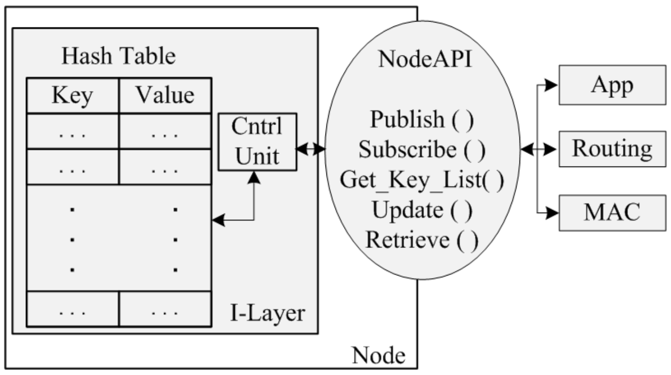


Figure 26. Layers sharing CL information in a publish-and-subscribe manner via I-Layer

verification succeeds, layer l can then call the *subscribe(key)* method. This will cause the control unit to remember that the entry with key k in the hash table will be read by layer l. After calling *subscribe(key)*, layer l will periodically check the I-Layer by calling *retrieve(key)* to gain the current value of the CL information it is interested in. Calling *retrieve(key)* in *NodeAPI* will tell the control unit of the I-Layer to check if layer l is among those layers that are interested in information associated with key k, and return the current value of the entry with key k in the hash table. Our implementation also supports subscribing to the CL information dynamically. If some layer wants to start subscribing to a new type of information at run time, it just needs to call *getKeyList()* first to verify the desired information is already published. The control unit of the I-Layer will cover the rest as described above.

5.4 Sample Scenario

5.4.1 Overview

In this section, we introduce a sample scenario developed based on CLAP. As mentioned in section 5.3.1, we consider the case of application control over CL interaction and protocol operation. The application layer monitors status info, published by MAC and routing layers, by subscribing to the I-Layer. The user is provided with an interface to both observe and specify values (i.e., run-time application preferences) of the status information currently being monitored (51). Upon receiving user input, the application layer processes user demands, based on which it issues control instructions to the underlying layers by publishing the instructions into the I-Layer. Underlying layers, which subscribe to control instructions published by the application layer, will take corresponding actions in an attempt to draw current status towards the user's/application's desired status. The details of this process are explained in the following subsections.

5.4.2 Monitoring and Control

In CLAP, the application's decision to modify the underlying protocol's behavior is based on the current status information, both local to the application layer (e.g., remaining battery) and that reported by other layers, and the run-time application/user preferences represented by target values for specific status information (e.g., delay, power consumption etc.). We define the set of target values $T^{'} = T^{'}_1, T^{'}_2, ..., T^{'}_i$. Each element of $T^{'}$ is assigned a weight W_i based on its priority among the other target elements, where $\Sigma_i W_i = 1$, and has a current value T_i and $T_i \in T$, where T is the set of current values of all elements in $T^{'}$. The distance between T_i and

$T_i^{'}$ is D_i which is computed as $D_i = W_i \frac{|T_i - T_i^{'}|}{T_i^{'}}$. Both W_i and $T_i^{'}$ are assigned by the application along with an error tolerance, tol_i. These assignments can be changed (reconfigured) according to the application demands and observed network operating conditions. We define a set of possible actions, A, the application can execute to modify underlying MAC and/or routing functionality where $A = \{A_1, A_2, ..., A_i\}$. Note that for presentation simplicity, we assume one action per status reported (T_i), however, each T_i can have several actions (A_is) affecting it, as shown in Table IX.

Without loss of generality, we assume the application is aware of the following:

- current average power consumption (T_1)
- current average delay (T_2)
- current average delivery ratio (T_3)
- desired target values of the above parameters (i.e., $\{T_1^{'}, T_2^{'}, T_3^{'}\}$) and their weights (i.e., $\{W_1, W_2, W_3\}$)

The application can modify routing and MAC operation via the following set of actions:

- retry limit (A_1)
- duty cycle (A_2)
- order packets in routing buffer based on destination (A_3)

Note that the above awareness info/actions can be extended or limited by the protocol designer.

We define delay as the time a packet spends in the routing buffer; delivery ratio is the ratio of number of packets successfully transmitted to the total number of packets sent. All monitored parameters (status info) are averaged over the same time interval (50 seconds in our simulations). Retry limit is the number of attempts by MAC layer to send a packet before dropping it, and duty cycle is the ratio between active period duration and total frame duration (i.e., *active* + *sleep*).

TABLE IX

DEMAND/ACTION RELATIONSHIP

Application demand	Related action
±*delivery ratio*	±*retry limit*
±*delay*	±*retry limit*; ±*duty cycle*
±*power consumption*	±*retry limit* ; ±*duty cycle*

5.4.3 Processing User Input

In the scope of the capabilities and awareness described in subsection 5.4.2, we explain the application-layer decision making process. The application is capable of increasing/decreasing the retry limit and/or duty cycle and/or reordering of packets in the routing buffer. The first two capabilities/actions have a direct effect on power consumption and delay; however, reordering packets does not. Increasing the retry limit is expected to increase power consumption and delay, however, it is also expected to increase delivery ratio, and vice versa. Increasing the duty cycle is expected to reduce delay and increase power consumption, and vice versa.

```
**Initialization**
1.  $i = 1$ /* $i$ loops over elements of a set */
2.  $j = 1$ /* $j$ loops over repetitions of the same action */
3.  set maximum values for $i$ and $j$ to $n$ and $m$
4.  save initial state   /* save $T'$ */
5.  set target values   /* set $T$ */
6.  set tol for each $D_i$ /* set tol $= 0$ for max accuracy */
**Phase 1: each $A_i$ individually**
7.  **Repeat**
8.      compute $D_i$
9.      if $(D_i \le tol_i)$
10.         $i++$
11.         go to line (8)
12.     else
13.         **Repeat**
14.             $D_i^{old} \leftarrow D_i$
15.             execute $A_i$   /* e.g. inc./dec. retry limit */
16.             if $((D_i \le tol_i) \,||\, (D_i^{old} \le D_i))$ /* target
                reached or unexpected outcome from $A_i$ */
17.                 $i++$
18.                 go to line (8)
19.             else
20.                 $j++$ /* controls no. of $A_i$ executions */
21.                 go to line (14)
22.         **Until** $j = m$
23. **Until** $i = n$
24. if $(D_1 \le tol_1 \&\& \ldots \&\& D_n \le tol_n)$
25.     terminate /* reached desired $T_i'$s */
26. else
27.     restore initial state
28.     go to phase 2
**Phase 2: All $A_i$s Simultaneously**
29. **Repeat**
30.     compute all $D_i$s
31.     $D_1^{old} \leftarrow D_1$, $D_2^{old} \leftarrow D_2$, …, $D_n^{old} \leftarrow D_n$
32.     execute $A_1, A_2, \ldots, A_n$
33.     if $(D_1 \le tol_1 \&\& \ldots \&\& D_n \le tol_n)$
34.         terminate /* reached desired $T_i'$s */
35.     elseif $(D_1 \le D_1^{old} \&\& \ldots \&\& D_n \le D_n^{old})$
36.         $j++$
37.         go to line (31)
38.     else
39.         exit /* unsuccessful */
40. **Until** $j = m$
```

Figure 27. Pseudo code of the decision making process in an application-controlled CL protocol

The relationship between the above status info (T) , user/application demands (T'), and possible actions (A), is greatly dependent on operating conditions (e.g., congestion, network topology, traffic patterns etc.). This makes quantification of such relationships infeasible. Therefore, the application adopts an empirical asymptotic approach to reach its goal while considering run-time operating conditions as illustrated in Figure 27. Note that this exhaustive search-like algorithm is implemented in the application layer and attempts to steer the functionality of underlying layers to meet application demands. However, there are no guarantees that the application demands will be achieved, and in such case the initial state (prior to applying the algorithm) is retrieved as illustrated in the pseudo code in Figure 27. The general relationship between application demands and related actions is illustrated in Table IX. Note that any combination of such demands is possible. Although any increase in retry limit is expected to increase duty cycle, we assume that the increase in retry count encountered by one packet is offset by a decrease in that of another packet sent in the same cycle, and hence consider both actions to be independent (mutually exclusive). This assumption is based on the MAC protocol capability of sending multiple packets/frame.

5.4.4 Simulation Results

We use CLAP to implement the sample scenario explained in Figure 27. The underlying MAC protocol is SMAC (26) and we assume that single schedule and routing information are available to nodes as in (10) (40). We simulate two networks, a 17-node cross-chain, and a 25-node grid. Each simulation lasts for 2000 seconds. On both networks, the distance between adjacent nodes is 200 meters. Important simulation parameters are listed in Table X.

TABLE X

SIMULATION PARAMETERS (CLAP)

Parameter	Value	Parameter	Value
Bandwidth	20kbps	*Comm. Range*	250m
RxPower	0.5W	*Interference Range*	550m
TxPower	0.5W	*DIFS*	10ms
IdlePower	0.45W	*SIFS*	5ms
SleepPower	0.05W	*Contention Window*	64ms
DataPckt	100B	*Cycle (SMAC)*	4544ms
ACK	10B	*Duty Cycle)*	15%

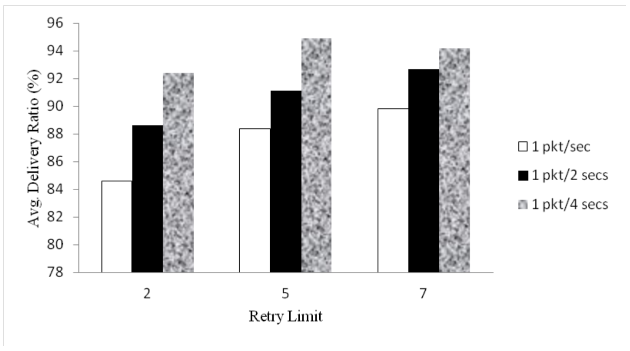


Figure 28. Average delivery ratio on the 17-node cross-chain network at various data rates and retry limits.

The main goal of our simulations is to demonstrate CLAP's ability of tailoring main performance metrics, namely, delivery ratio and delay. Energy consumption is kept constant by keeping a constant duty cycle throughout the simulations. Delay and delivery ratio were defined in section 5.4.2.

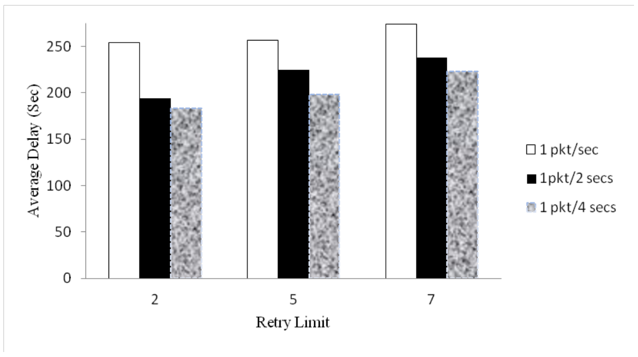


Figure 29. Average delay on the 17-node cross-chain network at various data rates and retry limits.

Figure 28 shows how increasing the retry limit can increase delivery ratio. When the data rate is at 1 packet/second and the retry limit is increased from 2 to 7, delivery ratio is increased from 84.6% to 90%. This modest 6.4% improvement is offset by a 7.8% increase in delay as shown in Figure 29. This delay penalty also offsets the delivery ratio gains at the lower data rates. This is due to the bottle neck occurring at the center node (at the crossing of the two chains) which greatly limits the traffic flow of both routes. Note that cross-chain results were averaged over the entire simulation duration and over the entire network.

Figure 30 shows the effect of CLAP's response to the user's (application's) demand of increasing delivery ratio by increasing the retry limit. The delivery ratio increases from 57% to 79% when the data rate is at 1 packet/second. This 38.6% increase in delivery ratio causes an increase in delay of only 16.8%, as shown in Figure 31. The least improvement in delivery ratio occurs at the lowest data rate (1 packet/ 4 seconds). This is due to the very small margin

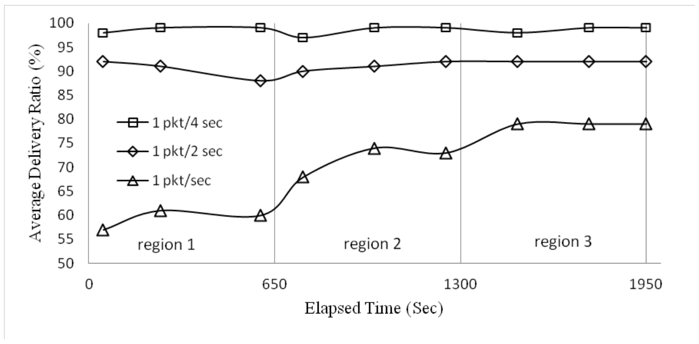


Figure 30. Average delivery ratio for the 25-node grid network. Retry limit is increased from 2 in region 1 to 5 in region 2, then to 7 in region 3

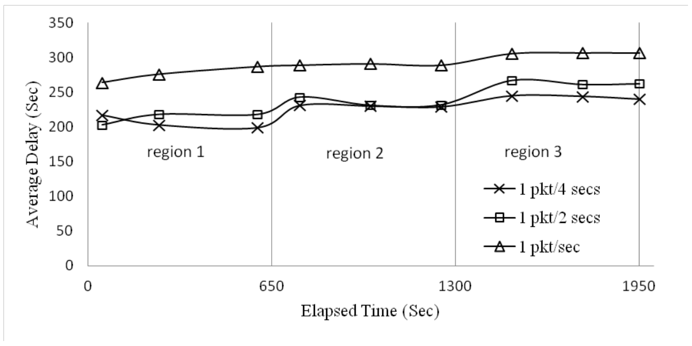


Figure 31. Average delay for the 25-node grid network. Retry limit is increased from 2 in region 1 to 5 in region 2, then to 7 in region 3.

available for any improvement in delivery ratio, as it is already at 98% when the retry limit is 2. This is reflected in the delay results at lower data rates, as shown in Figure 31.

5.5 Summary

The complexity of developing CL protocols in WSNs should not be affected by lack of CL information that is hidden in conventional single layered design approaches and their complementing simulators. Such conventional hierarchies influence the design process and add significant complexity to CL protocol implementation. Towards facilitating CL protocol design and simulation, we presented a new Cross-Layer Application-aware design Paradigm (CLAP). CLAP incorporates an I-Layer to efficiently moderate CL interactions. Any layer can access the I-Layer for monitoring/updating/control purposes. Not only this allows any combination of CL interactions to be realized by CLAP, it also grants the application layer new awareness and control capabilities. We integrated CLAP into SIDnet-SWANS (80) and simulated a sample scenario. The simulated scenario allows the user (application) to steer the operation of the underlying protocol(s), resulting in a reconfigurable CL application-controlled protocol.

CHAPTER 6

CONCLUSIONS AND FUTURE DIRECTIONS

In chapter 1, we introduced the main challenges faced and desirable features considered when designing and developing algorithms and protocols for Wireless Sensor Networks (WSNs). Particularly, we discussed resource constraints, adaptability/mobility, load balancing, scalability, heterogeneity, and reliability. Such challenges and desired characteristics pose various constraints in a wide array of WSN applications and are prioritized/jointly optimized based on the prospective application for which the protocol or algorithm is designed. In light of the above challenges and desired characteristics, we proposed several energy-efficient collaborative cross-layer schemes with various contexts of interest, each of which breaks new grounds for future directions.

In chapter 2, we studied utilizing routing layer information and inter-nodal collaborations to efficiently schedule multi-packet, multi-hop and multi-flow traffic patterns, while adapting to a wide range of traffic loads. We introduced CL-MAC, a Cross-Layer Medium Access Control protocol for synchronous heterogeneous WSNs. Distinct from other routing-oriented MAC protocols supporting construction of multi-hop flows, CL-MAC considers all pending packets in the routing layer buffer and all flow setup requests from neighbors, when setting up each flow. This allows CL-MAC to make more informed scheduling decisions, reflecting the current network status, and dynamically optimize its scheduling mechanism accordingly. CL-MAC has been tested on various heterogeneous networks imposing different traffic patterns and showed significant improvements over state of the art in this category of CL protocols. CL-MAC introduces new dimensions of awareness, particularly, network heterogeneity and multi-packet & multi-flow scheduling.

In chapter 3, we augmented CL interactions into a holistic model and investigated supporting multi-hop and multi-packet routing in asynchronous WSNs in order to improve end-to-end latency and overall power consumption. We proposed extending the knowledge of asynchronous MAC schemes and utilizing a combination of routing and duty-cycling information to temporarily synchronize nodes on the routing path between transmitter and receiver. Although the proposed technique is generic and can be applied to most asynchronous MAC schemes, we applied it to RI-MAC (Receiver Initiated MAC). Being one of the most efficient asynchronous MAC protocols due to its clever scheduling technique, RI-MAC was selected. We showed that the performance of RI-MAC is significantly improved after applying the new technique. Although the outcome of applying the proposed approach to other asynchronous MAC protocols is expected to yield different results, an improvement in latency is projected in all cases.

Chapter 4 discusses the collaboration between MAC and application layers to an realize an efficient context-aware data gathering solution. In this chapter, we discussed eliminating communication redundancy and improving load balancing in WSN data gathering applications. We introduced a CL Data-dependent Virtual Clustering-based Data Gathering technique (DVC-DG). The proposed technique integrates overhearing at the MAC layer with data being processed/communicated at upper layers to realize implicit virtual clusters. Unlike most existing clustering-based data gathering solutions, which employ explicit data-independent clustering techniques to choose cluster heads and members, DVC-DG eliminates the need for special cluster head election mechanisms and distributes common centralized cluster-head responsibilities over all cluster members. It also eliminates data redundancy at its very source rather than at the cluster-head. We also show that DVC-DG's performance is solely dependent on similarity in the sensed data and not on field correlations. While the two concepts of similarity and correlation are naturally tied in most applications, that is not always the case. In particular, random fields can benefit from DVC-DG. We show how DVC-DG can reduce communication redundancy and improve energy efficiency in various field similarity factors and different network setups. We also study the cost of extra overhearing and define the break-even point where the energy savings from message reduction is offset by the extra overhearing cost. This break-even point will vary among different MAC schemes as the cost of overhearing varies.

Finally, in chapter 5, we presented a Cross-Layer Application-aware Paradigm (CLAP) to facilitate CL design. CLAP utilizes a new means of exchanging and processing CL information, called the Information-Layer (I-Layer). The I-Layer facilitates collaboration across layers and between nodes and allows any layer in the network stack to impact the behavior of other layers, according to the context of interest. The I-Layer allows each layer of the stack to publish its local information to be shared with other layers and subscribe to other layers' shared information. We also augmented CLAP into the SIDnet-SWANS simulator via a new API design which eliminates the need for hacking/bypassing conventional design hierarchies and simulator architectures. This simplifies the design and implementation complexities of CL protocols. Furthermore, to demonstrate CLAPs unique capabilities, we utilize it to develop a sample CL protocol, which constantly monitors the application's current demands and reconfigures underlying protocols accordingly.

While each of the above solutions explores new dimensions of awareness that are exciting and promising, they also bring more interesting challenges. For example, maintaining synchronization is more challenging when multiple packets/flows are being scheduled in one cycle, especially as the network scales. In addition, the same challenges are expected when temporarily synchronizing nodes on a certain multi-hop path as discussed in chapter 3. This becomes increasingly important as the number of hops traversed between source and destination nodes increases.

The idea of suppressing redundant data is appealing from the perspective of reducing communication cost, however, it limits DVC-DG's application to specific types of data gathering applications where individual values of each node is not of interest (e.g. average value is sufficient). Modifying DVC-DG's operation to report similar data (instead of suppressing it) via short frames can significantly extend its applicability to virtually any data gathering application. An interesting idea is to replace the actual data frame with a short frame notifying the cluster head with the similarity. This will preserve all the data and will lead the cluster head to having complete knowledge of all member nodes' data without having to communicate the actual data.

CITED LITERATURE

1. Cody-Kenny, B., Guerin, D., Ennis, D., Simon Carbajo, R., Huggard, M., and Mc Goldrick, C.: Performance evaluation of the 6LoWPAN protocol on MICAz and TelosB motes. In Proceedings of the ACM workshop on Performance monitoring and measurement of heterogeneous wireless and wired networks, pages 25–30, 2009.

2. Pan, L., Wu, H., and Tzeng, N.-F.: An efficient and scalable prioritized mac protocol (pmac) for backbone communication in wireless sensor networks. In Sensor Technologies and Applications, 2009. SENSORCOMM '09. Third International Conference on, pages 508 –513, june 2009.

3. Suriyachai, P., Roedig, U., and Scott, A.: Implementation of a mac protocol for qos support in wireless sensor networks. In Pervasive Computing and Communications, 2009. PerCom 2009. IEEE International Conference on, pages 1 –6, march 2009.

4. Hightower, J. and Borriello, G.: Location systems for ubiquitous computing. Computer, 34:57–66, August 2001.

5. Li, J., Jannotti, J., De Couto, D. S. J., Karger, D. R., and Morris, R.: A scalable location service for geographic ad hoc routing. In Proceedings of the 6th annual international conference on Mobile computing and networking, MobiCom '00, pages 120–130, New York, NY, USA, 2000. ACM.

6. Lin, X., Kwok, Y., and Wang, H.: Cross-layer design for energy efficient communication in wireless sensor networks. Wireless Communications and Mobile Computing, 9(2):251–268, 2009.

7. Akyildiz, I., Melodia, T., and Chowdhury, K.: A survey on wireless multimedia sensor networks. Computer networks (Elsevier), 51(4):921–960, 2007.

8. Yahya, B. and Ben-Othman, J.: Towards a classification of energy aware mac protocols for wireless sensor networks. Wireless Communications and Mobile Computing, 9(12):1572 – 1607, 2009.

9. Demirkol, I., Ersoy, C., and Alagoz, F.: MAC protocols for wireless sensor networks: a survey. IEEE Communications Magazine, 44(4):115–121, 2006.

10. Du, S., Saha, A., and Johnson, D.: RMAC: A routing-enhanced duty-cycle MAC protocol for wireless sensor networks. In Proceedings of the IEEE International Conference on Computer Communications (INFOCOM), pages 1478–1486, May 2007.

11. Sun, Y., Du, S., Gurewitz, O., and Johnson, D.: DW-MAC: a low latency, energy efficient demand-wakeup MAC protocol for wireless sensor networks. In Proceedings of the ACM international symposium on Mobile ad hoc networking and computing (MobiHoc), pages 53–62, 2008.

12. Canli, T. and Khokhar, A.: PRMAC: Pipelined routing enhanced MAC protocol for wireless sensor networks. In Proceedings of the IEEE International Conference on Communications (ICC), pages 1–5, June 2009.

13. Kim, D. and Liu, M.: Optimal stochastic routing in low duty-cycled wireless sensor networks. In Proceedings of the 4th Annual International Conference on Wireless Internet, page 56. ICST (Institute for Computer Sciences, Social-Informatics and Telecommunications Engineering), 2008.

14. Lai, S. and Ravindran, B.: On distributed time-dependent shortest paths over duty-cycled wireless sensor networks. In INFOCOM, 2010 Proceedings IEEE, pages 1–9. IEEE, 2010.

15. Hefeida, M. S., Canli, T., and Khokhar, A.: Cl-mac: A cross-layer {MAC} protocol for heterogeneous wireless sensor networks. Ad Hoc Networks, 11(1):213 – 225, 2013.

16. Hefeida, M., Canli, T., and Khokhar, A.: Supporting multi-hop and multi-packet transmission in asynchronous wsns. In Wireless Days (WD), 2011 IFIP, pages 1 –6, oct. 2011.

17. Hefeida, M. and Khokhar, A.: A cross-layer approach for context-aware data gathering in wireless sensor networks. In Global Communications Conference (GLOBECOM), 2012 IEEE, pages 238–243, 2012.

18. Hefeida, M., Canli, T., Kshemkalyani, A., and Khokhar, A.: Context modeling in collaborative sensor network applications. In Collaboration Technologies and Systems (CTS), 2011 International Conference on, pages 274 –279, may 2011.

19. Hefeida, M., Shen, M., Kshemkalyani, A., and Khokhar, A.: Cross-layer protocols for wsns: A simple design and simulation paradigm. In Wireless Communications and Mobile Computing Conference (IWCMC), 2012 8th International, pages 844–849, 2012.

CITED LITERATURE (Continued)

20. Akyildiz, I., Su, W., Sankarasubramaniam, Y., and Cayirci, E.: A survey on sensor networks. IEEE Communications Magazine, 40(8):102–114, 2002.

21. Bachir, A., Dohler, M., Watteyne, T., and Leung, K.: MAC essentials for wireless sensor networks. IEEE Communications Surveys and Tutorials, 12(2):222 –248, quarter 2010.

22. Rossi, L., Krishnamachari, B., and Kuo, C.: Distributed parameter estimation for monitoring diffusion phenomena using physical models. In Proceedings of the IEEE Sensor and Ad Hoc Communications and Networks (SECON), pages 460–469, 2004.

23. Zhao, T. and Nehorai, A.: Distributed sequential Bayesian estimation of a diffusive source in wireless sensor networks. IEEE Transactions on Signal Processing, 55(4):1511–1524, 2007.

24. Bajwa, R., Rajagopal, R., Varaiya, P., and Kavaler, R.: In-pavement wireless sensor network for vehicle classification. In Proceedings of the IEEE International Conference on Information Processing in Sensor Networks (IPSN), 2011, pages 85–96.

25. Ye, W., Heidemann, J., and Estrin, D.: An energy-efficient mac protocol for wireless sensor networks. In Proceedings of the IEEE INFOCOM, volume 3, pages 1567–1576, 2002.

26. Ye, W., Heidemann, J., and Estrin, D.: Medium access control with coordinated adaptive sleeping for wireless sensor networks. IEEE/ACM Transactions on Networking, 12(3):493–506, 2004.

27. Joseph, P., Jason, H., and David, C.: Versatile low power media access for wireless sensor networks. In Proceedings of the 2nd international conference on Embedded networked sensor systems. Baltimore, MD, USA: ACM Press, 2004.

28. Van Dam, T. and Langendoen, K.: An adaptive energy-efficient MAC protocol for wireless sensor networks. In Proceedings of the ACM international conference on Embedded networked sensor systems (SenSys), pages 171–180, 2003.

29. Lin, P., Qiao, C., and Wang, X.: Medium access control with a dynamic duty cycle for sensor networks. In Proceedings of the IEEE Wireless Communications and Networking Conference (WCNC), volume 3, pages 1534 – 1539, 2004.

30. Arisha, K., Youssef, M., and Younis, M.: Energy-aware TDMA-based MAC for sensor networks. In System-Level Power Optimization for Wireless Multimedia Communication, eds. R. Karri and D. Goodman, pages 21–40. Springer US, 2002.

31. Ali, M., Suleman, T., and Uzmi, Z.: MMAC: a mobility-adaptive, collision-free MAC protocol for wireless sensor networks. In Proceedings of the IEEE IEEE International Performance, Computing, and Communications Conference (IPCCC), pages 401 – 407, April 2005.

32. Polastre, J., Hill, J., and Culler, D.: Versatile low power media access for wireless sensor networks. In Proceedings of the international conference on Embedded networked sensor systems (SenSys), 2004.

33. Buettner, M., Yee, G., Anderson, E., and Han, R.: X-MAC: a short preamble MAC protocol for duty-cycled wireless sensor networks. In Proceedings of the international conference on embedded networked sensor systems (SenSys), pages 307–320. ACM, 2006.

34. Wang, H., Zhang, X., Naït-Abdesselam, F., and Khokhar, A. A.: DPS-MAC: An asynchronous MAC protocol for wireless sensor networks. In Proceedings of the IEEE International Conference on High Performance Computing (HiPC), pages 393–404, 2007.

35. El-Hoiydi, A. and Decotignie, J.: WiseMAC: an ultra low power MAC protocol for the downlink of infrastructure wireless sensor networks. In Proceedings of the International Symposium on Computers and Communications (ISCC), volume 2, pages 244–251, 2004.

36. Ye, W., Silva, F., and Heidemann, J.: Ultra-low duty cycle MAC with scheduled channel polling. In Proceedings of the international conference on Embedded networked sensor systems (SenSys), pages 321–334, 2006.

37. Rhee, I., Warrier, A., Aia, M., Min, J., and Sichitiu, M.: Z-MAC: a hybrid MAC for wireless sensor networks. IEEE/ACM Transactions on Networking (TON), 16(3):511–524, 2008.

38. Rajendran, V., Obraczka, K., and Garcia-Luna-Aceves, J.: Energy-efficient, collision-free medium access control for wireless sensor networks. Wireless Networks (Springer), 12(1):63–78, 2006.

CITED LITERATURE (Continued)

39. Zhao, Y. Z., Nguyen, T., Ma, M., and Miao, C.: An energy-efficient MAC protocol with adaptive scheduling for wireless sensor networks. In Proceedings of the IEEE International Conference on Industrial Informatics (INDIN), pages 446–451, June 2009.

40. Canli, T., Hefeida, M., and Khokhar, A.: BulkMAC: A cross-layer based MAC protocol for wireless sensor networks. In Proceedings of the International Wireless Communications and Mobile Computing Conference (IWCMC), pages 442–446, 2010.

41. Kim, J., Park, K., and Park, D.: An energy-efficient scheduling MAC protocol for wireless sensor networks. In Proceedings of the IEEE Consumer Communications and Networking Conference (CCNC), pages 655–659, 2007.

42. Zhang, X., Ansari, J., and Mähönen, P.: Traffic aware medium access control protocol for wireless sensor networks. In Proceedings of the ACM international symposium on Mobility management and wireless access, pages 140–148, 2009.

43. Nguyen, K. and Ji, Y.: Am-mac: an energy efficient, adaptive multi-hop mac protocol for sensor networks. In Proceedings of the International Wireless Communications and Mobile Computing Conference (IWCMC), pages 432–436, New York, NY, USA, 2010. ACM.

44. UC Berkeley, LBL, USC/ISI and Xerox Parc, NS-2 documentation and software, Version 2.29, October 2005.

45. Melodia, T., Vuran, M., and Pompili, D.: The state of the art in cross-layer design for wireless sensor networks. Wireless Systems and Network Architectures in Next Generation Internet, pages 78–92, 2006.

46. Srivastava, V. and Motani, M.: Cross-layer design: a survey and the road ahead. Communications Magazine, IEEE, 43(12):112–119, 2005.

47. Conti, M., Maselli, G., Turi, G., and Giordano, S.: Cross-layering in mobile ad hoc network design. Computer, 37(2):48 – 51, feb 2004.

48. Lott, C. and Teneketzis, D.: Stochastic routing in ad-hoc networks. Automatic Control, IEEE Transactions on, 51(1):52–70, 2006.

CITED LITERATURE (Continued)

49. Sun, Y., Gurewitz, O., and Johnson, D.: RI-MAC: a receiver-initiated asynchronous duty cycle MAC protocol for dynamic traffic loads in wireless sensor networks. In Proceedings of the 6th ACM conference on Embedded network sensor systems, pages 1–14. ACM New York, NY, USA, 2008.

50. McCanne, S. and Floyd, S.: ns network simulator–version 2. URL: http://www. isi. edu/nsnam/ns.

51. Hefeida, M., Canli, T., Kshemkalyani, A., and Khokhar, A.: Context modeling in collaborative sensor network applications. In Collaboration Technologies and Systems (CTS), 2011 International Conference on, pages 274–279. IEEE.

52. Hu, C., Kim, H., and Hou, J.: An analysis of the binary exponential backoff algorithm in distributed mac protocols. Technical report, 2005.

53. Younis, O., Krunz, M., and Ramasubramanian, S.: Node clustering in wireless sensor networks: recent developments and deployment challenges. Network, IEEE, 20(3):20 – 25, may-june 2006.

54. Heinzelman, W., Chandrakasan, A., and Balakrishnan, H.: Energy-efficient communication protocol for wireless microsensor networks. In System Sciences, 2000. Proceedings of the 33rd Annual Hawaii International Conference on, page 10 pp. vol.2, jan. 2000.

55. Younis, O. and Fahmy, S.: Heed: a hybrid, energy-efficient, distributed clustering approach for ad hoc sensor networks. Mobile Computing, IEEE Transactions on, 3(4):366 – 379, oct.-dec. 2004.

56. Vuran, M. and Akyildiz, I.: Spatial correlation-based collaborative medium access control in wireless sensor networks. Networking, IEEE/ACM Transactions on, 14(2):316 – 329, april 2006.

57. Jamieson, K., Balakrishnan, H., and Tay, Y.: Sift: A MAC protocol for event-driven wireless sensor networks. Lecture Notes in Computer Science, 3868:260, 2006.

58. Iima, Y., Kanzaki, A., Hara, T., and Nishio, S.: Overhearing-based data transmission reduction for periodical data gathering in wireless sensor networks. In Complex, Intelligent and Software Intensive Systems, 2009. CISIS '09. International Conference on, pages 1048 –1053, march 2009.

CITED LITERATURE (Continued)

59. Cano, C., Bellalta, B., Sfairopoulou, A., Oliver, M., and Barceló, J.: Taking advantage of overhearing in low power listening wsns: A performance analysis of the lwt-mac protocol. Mob. Netw. Appl., 16:613–628, October 2011.

60. Chen, Z. and Khokhar, A.: Self organization and energy efficient TDMA MAC protocol by wake up for wireless sensor networks. In IEEE SECON, volume 2004, pages 210–207, 2004.

61. Lin, C. and Gerla, M.: Adaptive clustering for mobile wireless networks. Selected Areas in Communications, IEEE Journal on, 15(7):1265 –1275, sep 1997.

62. Banerjee, S. and Khuller, S.: A clustering scheme for hierarchical control in multi-hop wireless networks. In INFOCOM 2001. Twentieth Annual Joint Conference of the IEEE Computer and Communications Societies. Proceedings. IEEE, volume 2, pages 1028 –1037 vol.2, 2001.

63. Basagni, S.: Distributed clustering for ad hoc networks. In Parallel Architectures, Algorithms, and Networks, 1999. (I-SPAN '99) Proceedings. Fourth InternationalSymposium on, pages 310 –315, 1999.

64. Chatterjee, M., Das, S. K., and Turgut, D.: Wca: A weighted clustering algorithm for mobile ad hoc networks. Cluster Computing, 5:193–204, 2002. 10.1023/A:1013941929408.

65. Deosarkar, B., Yadav, N., and Yadav, R.: Clusterhead selection in clustering algorithms for wireless sensor networks: A survey. In Computing, Communication and Networking, 2008. ICCCn 2008. International Conference on, pages 1 –8, dec. 2008.

66. Heinzelman, W., Chandrakasan, A., and Balakrishnan, H.: An application-specific protocol architecture for wireless microsensor networks. Wireless Communications, IEEE Transactions on, 1(4):660 – 670, oct 2002.

67. Blum, R. and Sadler, B.: Energy efficient signal detection in sensor networks using ordered transmissions. Signal Processing, IEEE Transactions on, 56(7):3229–3235, 2008.

68. Hesham, L., Sultan, A., Nafie, M., and Digham, F.: Cooperative sensing with sequential ordered transmissions to secondary fusion center. In Acoustics, Speech and Signal Processing (ICASSP), 2011 IEEE International Conference on, pages 2988 –2991, may 2011.

69. Zou, Q., Zheng, S., and Sayed, A.: Cooperative spectrum sensing via sequential detection for cognitive radio networks. In Signal Processing Advances in Wireless Communications, 2009. SPAWC'09. IEEE 10th Workshop on, pages 121–125. IEEE, 2009.

70. Kwak, B.-J., Song, N.-O., and Miller, L.: Performance analysis of exponential backoff. IEEE/ACM Transactions on Networking, 13(2):343 – 355, april 2005.

71. Ramakrishnan, M. and Ranjan, P.: Adaptive power control with overhearing avoidance for wireless sensor networks. In Communication Systems and Networks (COMSNETS), 2010 Second International Conference on, pages 1 –8, jan. 2010.

72. Vuran, M. and Akyildiz, I.: Xlp: A cross-layer protocol for efficient communication in wireless sensor networks. IEEE Transactions on Mobile Computing, pages 1578–1591, 2010.

73. Cui, S., Madan, R., Goldsmith, A., and Lall, S.: Joint routing, mac, and link layer optimization in sensor networks with energy constraints. In Communications, 2005. ICC 2005. 2005 IEEE International Conference on, volume 2, pages 725–729. IEEE, 2005.

74. Winter, R., Schiller, J., Nikaein, N., and Bonnet, C.: Crosstalk: Cross-layer decision support based on global knowledge. Communications Magazine, IEEE, 44(1):93–99, 2006.

75. Knopp, R., Nikaein, N., Bonnet, C., Aiache, H., Conan, V., Masson, S., Guibe, G., and Martret, C.: Overview of the widens architecture, a wireless ad hoc network for public safety. In IEEE SECON 2004, 2004.

76. Conti, M., Maselli, G., Turi, G., and Giordano, S.: Cross-layering in mobile ad hoc network design. Computer, 37(2):48–51, 2004.

77. Safwat, A.: A novel framework for cross-layer design in wireless ad hoc and sensor networks. In Global Telecommunications Conference Workshops, 2004. GlobeCom Workshops 2004. IEEE, pages 130–135. IEEE, 2004.

78. Aune, F.: Cross-layer design tutorial. Norwegian University of Science and Technology, Dept. of Electronics and Telecommunications, 2004.

79. Akyildiz, I., Vuran, M., and Akan, O.: A cross-layer protocol for wireless sensor networks. In Information Sciences and Systems, 2006 40th Annual Conference on, pages 1102–1107. Ieee, 2006.

80. Ghica, O., Trajcevski, G., Scheuermann, P., Bischof, Z., and Valtchanov, N.: Sidnet-swans: a simulator and integrated development platform for sensor networks applications. In Proceedings of the 6th ACM conference on Embedded network sensor systems, pages 385–386. ACM, 2008.

81. Adve, S., Harris, A., Hughes, C., Jones, D., Kravets, R., Nahrstedt, K., Sachs, D., Sasanka, R., Srinivasan, J., and Yuan, W.: The illinois grace project: Global resource adaptation through cooperation. In Proc. of Workshop on Self-Healing, Adaptive and self-MANaged Systems. Citeseer, 2002.

82. Chen, K., Shah, S., and Nahrstedt, K.: Cross-layer design for data accessibility in mobile ad hoc networks. Wireless Personal Communications, 21(1):49–76, 2002.

83. Iima, Y., Kanzaki, A., Hara, T., and Nishio, S.: Overhearing-based data transmission reduction for periodical data gathering in wireless sensor networks (pdf). 2009.

84. Henriksen, J.: Taming the complexity dragon. Journal of Simulation, 2(1):3–17, 2008.

85. Kawadia, V. and Kumar, P.: A cautionary perspective on cross-layer design. Wireless Communications, IEEE, 12(1):3–11, 2005.

86. Lee, L.: Cross-layer design and optimization for wireless sensor networks. Technical report, DTIC Document, 2006.

87. Garcia-Macias, J. and Gomez, J.: Manet versus wsn. Sensor Networks and Configuration, pages 369–388, 2007.

88. Barr, R. and et al.: Jist: Embedding simulation time into a virtual machine. In IN EUROSIM CONGRESS ON MODELLING AND SIMULATION, 2004.

89. Varga, A. and Hornig, R.: An overview of the omnet++ simulation environment. In Proceedings of the 1st international conference on Simulation tools and techniques for communications, networks and systems & workshops, Simutools '08, pages 60:1–60:10, ICST, Brussels, Belgium, Belgium, 2008. ICST (Institute for Computer Sciences, Social-Informatics and Telecommunications Engineering).

CITED LITERATURE (Continued)

90. Zeng, X., Bagrodia, R., and Gerla, M.: Glomosim: a library for parallel simulation of large-scale wireless networks. SIGSIM Simul. Dig., 28(1):154–161, July 1998.

91. ur Rehman Khan, A., Bilal, S., and Othman, M.: A performance comparison of open source network simulators for wireless networks. In Control System, Computing and Engineering (ICCSCE), 2012 IEEE International Conference on, pages 34–38, 2012.

92. Schoch, E., Feiri, M., Kargl, F., and Weber, M.: Simulation of ad hoc networks: ns-2 compared to jist/swans. In Proceedings of the 1st international conference on Simulation tools and techniques for communications, networks and systems & workshops, pages 1–8. ICST (Institute for Computer Sciences, Social-Informatics and Telecommunications Engineering), 2008.

93. Weingärtner, E., Vom Lehn, H., and Wehrle, K.: A performance comparison of recent network simulators. In Proceedings of the 2009 IEEE international conference on Communications, ICC'09, pages 1287–1291, Piscataway, NJ, USA, 2009. IEEE Press.

94. Kargl, F. and Schoch, E.: Simulation of manets: a qualitative comparison between jist/swans and ns-2. In Proceedings of the 1st international workshop on System evaluation for mobile platforms, MobiEval '07, pages 41–46, New York, NY, USA, 2007. ACM.

VITA

Name: Mohamed Salem Hefeida

Education: Ph.D., Electrical and Computer Engineering, University of Illinois at Chicago, Chicago, Illinois, 2013

M.Sc., Electronics and Communications Engineering, Arab Academy for Science & Technology and Maritime Transport, Alexandria, Egypt, 2006

B.Sc., Electronics and Communications Engineering, Arab Academy for Science & Technology and Maritime Transport, Alexandria, Egypt, 2004

Experience: Instructor, University of Illinois at Chicago, *Spring 2013-*

Research Assistant at Scalable Parallel Algorithms and Multimedia Systems Laboratory, University of Illinois at Chicago, *Fall 2010-*

Teaching Assistant, ECE Department, University of Illinois at Chicago, *Fall 2007-Fall 2012*

Project Lead, EduSerc/Northrop Grumman, *Summers 2009-11*

Honors: Dean's Scholar Award: University of Illinois at Chicago (UIC), *Spring 2013.*

AAAEA Scholarship: Arab American Association for Engineers and Architects, *Fall 2012.*

ECE Department Representative TA: University of Illinois at Chicago (UIC), *Fall 2010 and 2011.*

Outstanding Student Award: Arab Academy for Science and Technology (AAST), *Fall 2004.*

Undergrad Valedictorian Scholarship: Arab Academy for Science and Technology (AAST), *Spring 2000 - Spring 2004.*